

PIC18F97J60 系列 数据手册

带有以太网模块的 64/80/100 引脚高性能 1 Mb 闪存单片机

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 "牢不可破"的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了 《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip 位于美国亚利桑那州 Chandler 和 Tempe、位于俄勒冈州 Gresham 及位于加利福尼亚州 Mountain View 的全球总部、设计中心和晶圆生产厂均通过了 ISO/TS-16949:2002 认证。公司在 PIC® 8 位单片机、 KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品方面的质量体系流程均符合 ISO/TS-16949:2002。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。



PIC18F97J60 系列

带有以太网模块的 64/80/100 引脚 高性能 1 Mb 闪存单片机

以太网特性：

- IEEE 802.3 兼容的以太网控制器
- 集成 MAC 和 10Base-T PHY
- 8 KB 发送 / 接收数据包缓冲区 SRAM
- 支持一个带自动极性检测和校正的 10Base-T 端口
- 在发生冲突时可编程自动重发
- 可编程填充和 CRC 生成
- 可编程自动拒绝错误数据包
- 可驱动 2 个 LED 指示器的活动输出信号
- 缓冲器：
  - 可配置发送 / 接收缓冲区大小
  - 硬件管理的循环接收 FIFO
  - 字节宽度的随机访问和顺序访问
  - 用于快速数据传送的内部 DMA
  - 支持各种协议的硬件校验和计算
- MAC：
  - 支持单播、组播和广播数据包
  - 多达 64 字节的可编程模式匹配 （偏移量可由用户定义）
  - 多个数据包格式的可编程唤醒
- PHY：
  - 整形输出滤波器
  - 环回模式

灵活的振荡器结构：

- 可选择的系统时钟，来自单个 25 MHz 外部时钟源：
  - 2.78 到 41.67 MHz
- 内部 31 kHz 振荡器
- 辅助振荡器使用 Timer1 （工作频率为 32 kHz）
- 故障保护时钟监视器：
  - 当振荡器停止工作时可使器件安全断电
- 双速振荡器启动

外部存储器总线 （仅限 100 引脚器件）：

- 地址容量最高可达 2 MB
- 8 位或 16 位接口
- 12 位、 16 位和 20 位寻址模式

外设特点：

- 高灌 / 拉电流：在 PORTB 和 PORTC 上为 25 mA/25 mA
- 5 个定时器模块 （Timer0 到 Timer4）
- 4 个外部中断引脚
- 2 个捕捉 / 比较 /PWM （CCP）模块
- 3 个增强型捕捉 / 比较 /PWM （ECCP）模块：
  - 1、 2 或 4 路 PWM 输出
  - 可选择极性
  - 可编程死区延时
  - 自动关闭和自动重启
- 多达 2 个主控同步串行口（Master Synchronous Serial Port， MSSP）模块，支持 SPI （所有 4 种模式）和 I^2C™ 主 / 从模式
- 多达 2 个增强型 USART 模块：
  - 支持 RS-485、 RS-232 和 LIN 1.2
  - 启动位自动唤醒
  - 自动波特率检测
- 多达 16 路通道的 10 位模数转换器模块 （A/D）：
  - 自动采样功能
  - 可在休眠模式下进行转换
- 输入复用的双模拟比较器
- 并行从动端口 （Parallel Slave Port， PSP）模块 （仅限 100 引脚器件）

单片机特性：

- 最大容许输入电压为 5.5V （仅限数字引脚）
- 低功耗、高速 CMOS 闪存技术：
  - 可在软件控制下自行重新编程
- 为重入代码优化的 C 编译器架构
- 功耗管理特性：
  - 运行：CPU 工作，外设打开
  - 空闲：CPU 不工作，外设打开
  - 休眠：CPU 不工作，外设关闭
- 中断优先级
- 8 x 8 单周期硬件乘法器
- 扩展型看门狗定时器 （Watchdog Timer， WDT）：
  - 可编程周期从 4 ms 到 134s
- 通过两个引脚进行单电源 3.3V 在线串行编程 （In-Circuit Serial Programming™， ICSP™)
- 通过两个引脚进行带有 3 个断点的在线调试 （In-Circuit Debug， ICD）
- 工作电压范围为 2.35V 到 3.6V（使用以太网模块时为 3.14V 到 3.45V）
- 片上 2.5V 稳压器

器件	闪存程序存储器（字节）	SRAM 数据存储器（字节）	以太网 TX/RX 缓冲区（字节）	I/O	10 位 A/D（通道）	CCP/ECCP	MSSP			EUSART	比较器	8/16 位定时器	PSP	外部存储器总线
								SPI	I²C™					
PIC18F66J60	64K	3808	8192	39	11	2/3	1	有	有	1	2	2/3	无	无
PIC18F66J65	96K	3808	8192	39	11	2/3	1	有	有	1	2	2/3	无	无
PIC18F67J60	128K	3808	8192	39	11	2/3	1	有	有	1	2	2/3	无	无
PIC18F86J60	64K	3808	8192	55	15	2/3	1	有	有	2	2	2/3	无	无
PIC18F86J65	96K	3808	8192	55	15	2/3	1	有	有	2	2	2/3	无	无
PIC18F87J60	128K	3808	8192	55	15	2/3	1	有	有	2	2	2/3	无	无
PIC18F96J60	64K	3808	8192	70	16	2/3	2	有	有	2	2	2/3	有	有
PIC18F96J65	96K	3808	8192	70	16	2/3	2	有	有	2	2	2/3	有	有
PIC18F97J60	128K	3808	8192	70	16	2/3	2	有	有	2	2	2/3	有	有



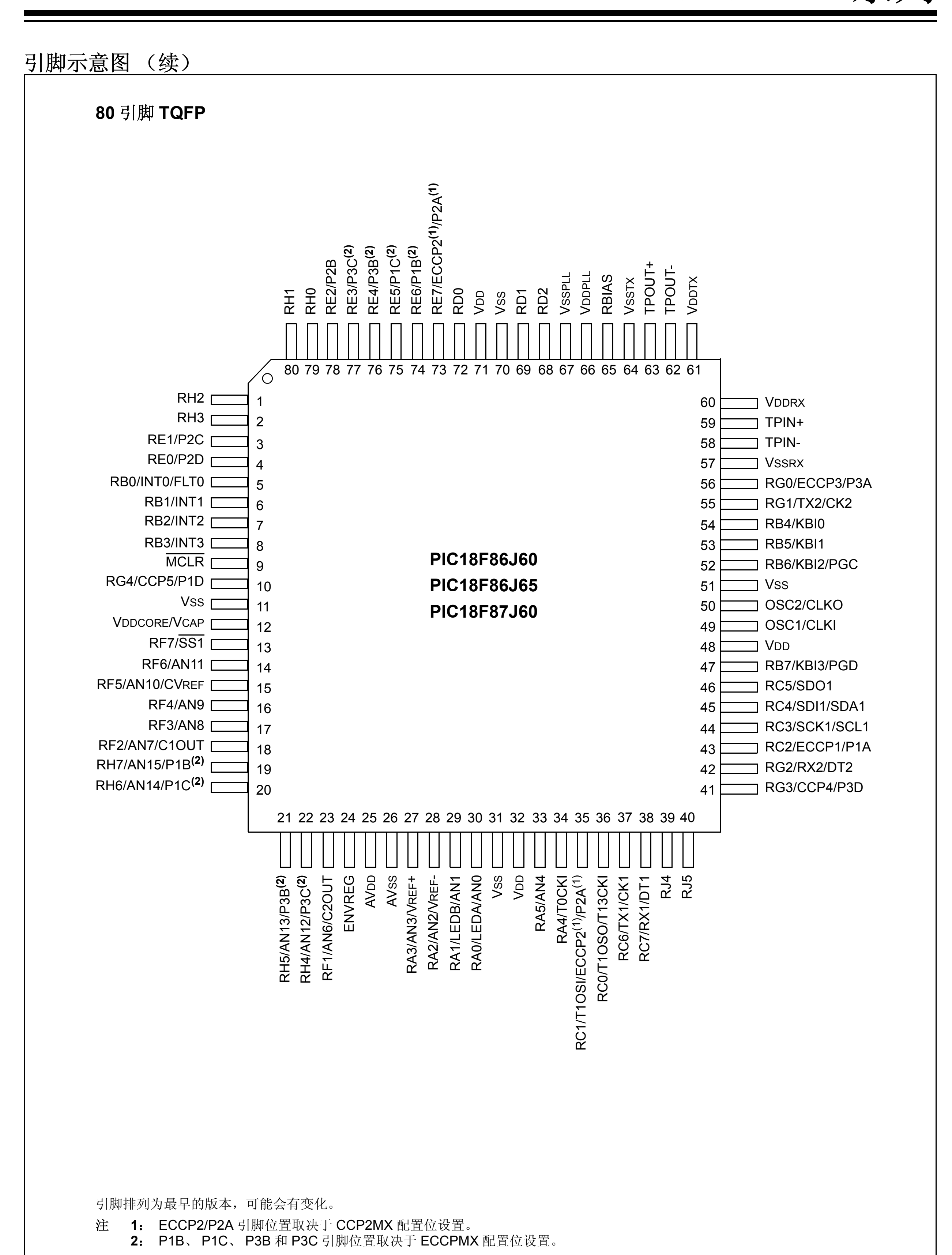
引脚示意图

64 引脚 TQFP

PIC18F66J60
PIC18F66J65
PIC18F67J60

1 RE1/P2C
2 RE0/P2D
3 RB0/INT0/FLT0
4 RB1/INT1
5 RB2/INT2
6 RB3/INT3
7 $\overline{\text{MCLR}}$
8 RG4/CCP5/P1D
9 VSS
10 VDDCORE/VCAP
11 RF7/$\overline{\text{SS1}}$
12 RF6/AN11
13 RF5/AN10/CVREF
14 RF4/AN9
15 RF3/AN8
16 RF2/AN7/C1OUT
17 RF1/AN6/C2OUT
18 ENVREG
19 AVDD
20 AVSS
21 RA3/AN3/VREF+
22 RA2/AN2/VREF-
23 RA1/LEDB/AN1
24 RA0/LEDA/AN0
25 VSS
26 VDD
27 RA5/AN4
28 RA4/T0CKI
29 RC1/T1OSI/ECCP2/P2A
30 RC0/T1OSO/T13CKI
31 RC6/TX1/CK1
32 RC7/RX1/DT1
33 RC2/ECCP1/P1A
34 RC3/SCK1/SCL1
35 RC4/SDI1/SDA1
36 RC5/SDO1
37 RB7/KBI3/PGD
38 VDD
39 OSC1/CLKI
40 OSC2/CLKO
41 VSS
42 RB6/KBI2/PGC
43 RB5/KBI1
44 RB4/KBI0
45 VSSRX
46 TPIN-
47 TPIN+
48 VDDRX
49 VDDTX
50 TPOUT-
51 TPOUT+
52 VSSTX
53 RBIAS
54 VDDPLL
55 VSSPLL
56 VSS
57 VDD
58 RD2/CCP4/P3D
59 RD1/ECCP3/P3A
60 RD0/P1B
61 RE5/P1C
62 RE4/P3B
63 RE3/P3C
64 RE2/P2B

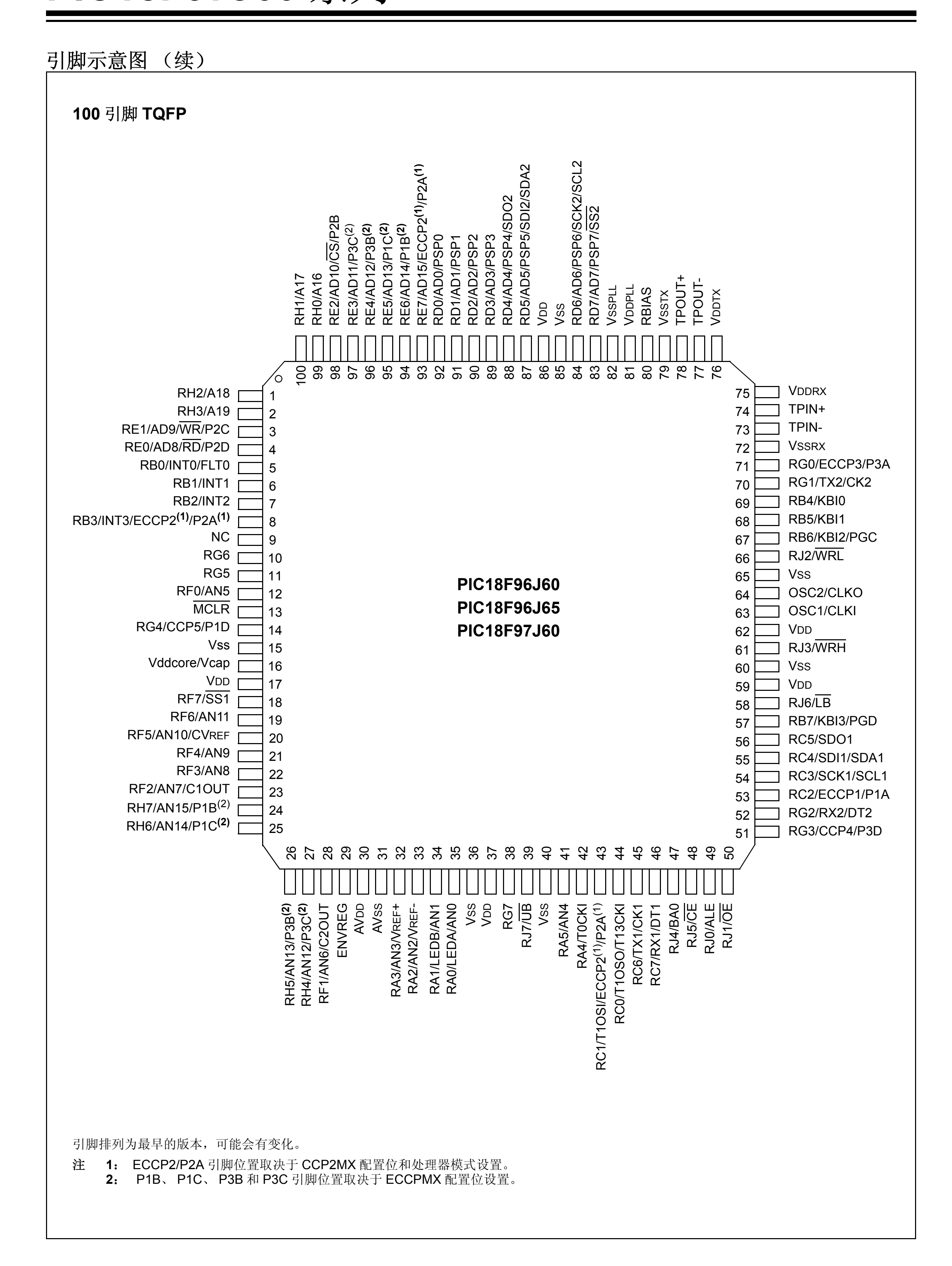
引脚示意图 （续）



引脚排列为最早的版本，可能会有变化。

注 1： ECCP2/P2A 引脚位置取决于 CCP2MX 配置位设置。
2： P1B、 P1C、 P3B 和 P3C 引脚位置取决于 ECCPMX 配置位设置。

引脚示意图 （续）



引脚排列为最早的版本，可能会有变化。

注 1： ECCP2/P2A 引脚位置取决于 CCP2MX 配置位和处理器模式设置。
2： P1B、 P1C、 P3B 和 P3C 引脚位置取决于 ECCPMX 配置位设置。

 超前信息

目录

致客户

我们旨在提供最佳文档供客户正确使用 Microchip 产品。为此，我们将不断改进出版物的内容和质量，使之更好地满足您的要求。出版物的质量将随新文档及更新版本的推出而得到提升。

如果您对本出版物有任何问题和建议，请通过电子邮件联系我公司 TRC 经理，电子邮件地址为 **CTRC@microchip.com**，或将本数据手册后附的 《**读者反馈表**》传真到 86-21-5407 5066。我们期待您的反馈。

最新数据手册

欲获得本数据手册的最新版本，请查询我公司的网站：

http://www.microchip.com

查看数据手册中任意一页下边角处的文献编号即可确定其版本。文献编号中数字串后的字母是版本号，例如：DS30000A 是 DS30000 的 A 版本。

勘误表

现有器件可能带有一份勘误表，描述了实际运行与数据手册中记载内容之间存在的细微差异以及建议的变通方法。一旦我们了解到器件 / 文档存在某些差异时，就会发布勘误表。勘误表上将注明其所适用的硅片版本和文件版本。

欲了解某一器件是否存在勘误表，请通过以下方式之一查询：

- Microchip 网站 http://www.microchip.com
- 当地 Microchip 销售办事处 （见最后一页）

在联络销售办事处时，请说明您所使用的器件型号、硅片版本和数据手册版本 （包括文献编号）。

客户通知系统

欲及时获知 Microchip 产品的最新信息，请到我公司网站 **www.microchip.com** 上注册。

1.0 器件概述

本文档涉及以下器件的具体信息：

- PIC18F66J60
- PIC18F66J65
- PIC18F67J60
- PIC18F86J60
- PIC18F86J65
- PIC18F87J60
- PIC18F96J60
- PIC18F96J65
- PIC18F97J60

PIC18F97J60 是新面世的低压产品系列，继承了 PIC18 单片机的传统优点（即出色的计算性能、丰富的功能集以及极具竞争力的价格）。这些特点使得 PIC18F97J60 系列成为许多高性能，尤其是那些成本作为首要考虑因素的应用的理想选择。

1.1 内核特性

1.1.1 纳瓦技术

PIC18F97J60 系列的所有器件具有一系列能在工作时显著降低功耗的功能。主要包含以下几项：

- **备用运行模式：**通过将 Timer1 或内部 RC 振荡器作为单片机时钟源，可使代码执行时的功耗降低大约 90%。
- **多种空闲模式：**单片机还可工作在其 CPU 内核禁止而外设仍然工作的情况下。处于这些状态时，功耗能降得更低，只有正常工作时的 4%。
- **动态模式切换：**在器件工作期间可由用户代码调用功耗管理模式，允许用户将节能的理念融入到他们的应用软件设计中。

1.1.2 振荡器选项和特性

PIC18F97J60 系列的所有器件可提供 5 个不同的振荡器选项，使用户在开发应用硬件时有很大的选择范围。这些选择包括：

- 2 种晶振模式，使用晶振或陶瓷谐振器。
- 2 种外部时钟模式，提供 4 分频时钟输出选项。
- 1 个锁相环 （Phase Lock Loop， PLL）倍频器，可在外部振荡器模式下使用，可使时钟速度最高达到 41.67 MHz。
- 具有固定的 31 kHz 输出的 RC 振荡器，为对时序不敏感的应用提供一个极低功耗的选择。

内部振荡器模块还提供了一个稳定的参考源，增加了以下功能以使器件更安全地工作：

- **故障保护时钟监视器：**该部件持续监视主时钟源，将其与内部振荡器提供的参考信号作比较。如果时钟发生了故障，单片机会将时钟源切换到内部振荡器，使器件可继续低速工作或安全地关闭应用。
- **双速启动：**该功能允许在上电复位或从休眠模式唤醒时将内部振荡器用作时钟源，直到主时钟源可用为止。

1.1.3 扩展存储器

PIC18F97J60 系列为应用程序代码提供了充足的空间——从 64 KB 到 128 KB。程序存储器的闪存单元经测评最多可承受 100 次擦写。在不刷新的情况下，数据保存期保守地估计在 20 年以上。

PIC18F97J60 系列还为动态应用程序数据提供了充足的空间——3808 字节的数据 RAM。

1.1.4 外部存储器总线

即使万一 （这种情况很少发生） 128 KB 的存储器对于某种应用不够用， PIC18F97J60 系列的 100 引脚器件还实现了外部存储器总线。这使单片机的内部程序计数器的寻址空间可以扩展到高达 2 MB，为用户提供 8 位器件很少能达到的数据访问水平。这使得有更多的存储器选项可供选择，包括：

- 片上和外部存储器结合使用，存储容量最多可达 2 MB
- 使用外部闪存来存储可再编程应用程序代码或大数据表
- 使用外部 RAM 器件存储大量的变量数据

1.1.5 扩展指令集

PIC18F97J60 系列在 PIC18 指令集的基础上进行了可选择的扩展，添加了 8 条新指令和一个变址寻址模式。此扩展可以使用一个器件配置选项使能，它是为优化重入应用程序代码而特别设计的，这些代码原来是使用高级语言 （如 C 语言）开发的。

1.1.6 移植方便

无论存储器容量如何，所有器件都共享相同的丰富外设，使应用程序的扩展和升级工作变得轻松而容易。

1.2 其他特殊功能

- **通信：**PIC18F97J60 系列包括一系列串行通信外设，包括多至 2 个的独立增强型 USART 和多至 2 个的主控 SSP 模块，能够进行 SPI 和 I^{2}C™ （主控和从动）模式操作。此外，通用 I/O 端口中有一个可重新配置为 8 位并行从动端口，用于进行直接的处理器间通信。
- **CCP 模块：**本系列的所有器件都包括 2 个捕捉 / 比较 /PWM (CCP) 模块和 3 个增强型 CCP （ECCP）模块，可方便灵活地实现控制应用。在同一时间，最多可以使用 4 种不同时基来执行几项不同的操作。 3 个 ECCP 模块中的每一个最多都可提供 4 路 PWM 输出，从而总共可有 12 路 PWM。ECCP 模块还提供许多有用功能，包括极性选择、可编程死区延时、自动关闭与重启，以及半桥与全桥输出模式。
- **10 位 A/D 转换器 ：**该模块具备可编程采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，因而减少了代码开销。
- **扩展型看门狗定时器 （WDT）：**该增强型版本加入了一个 16 位预分频器，可以保证扩展的超时范围在工作电压和温度变化时保持稳定。超时周期请参见**第 27.0 节 “电气特性”**。

1.3 系列中各产品的详细说明

PIC18F97J60 系列器件以 64 引脚、 80 引脚和 100 引脚封装形式提供。图 1-1、图 1-2 和图 1-3 分别为这三类器件的框图。

这些器件在以下四个方面存在差异：

1. 闪存程序存储器（3种大小，范围从PIC18FX6J60 器件的 64 KB 到 PIC18FX7J60 器件的 128 KB）。
2. A/D 通道数（64 引脚器件有 11 路，80 引脚器件有 15 路， 100 引脚器件有 16 路）。
3. 串行通信模块数（64 引脚器件上有 1 个 EUSART 模块和 1 个 MSSP 模块， 80 引脚器件上有 2 个 EUSART 模块和 1 个 MSSP 模块， 100 引脚器件上有 2 个 EUSART 模块和 2 个 MSSP 模块）。
4. I/O 引脚数（64 引脚器件上 39 个，80 引脚器件上 55 个， 100 引脚器件上 70 个）。

本系列器件的所有其他功能都是相同的。表 1-1、表 1-2 和表 1-3 中总结了这些功能。

表 1-4、表 1-5 和表 1-6 给出了本系列中所有器件的引脚说明。

表 1-1： PIC18F97J60 系列器件特性 （64 引脚器件）

特性	PIC18F66J60	PIC18F66J65	PIC18F67J60
工作频率	DC – 41.67 MHz	DC – 41.67 MHz	DC – 41.67 MHz
程序存储器 （字节数）	64K	96K	128K
程序存储器 （指令数）	32764	49148	65532
数据存储器 （字节数）	3808		
中断源	26		
I/O 端口	端口 A, B, C, D, E, F, G		
I/O 引脚	39		
定时器	5		
捕捉 / 比较 /PWM 模块	2		
增强型捕捉 / 比较 /PWM 模块	3		
串行通信	MSSP (1) 和增强型 USART (1)		
以太网通信 （10Base-T）	有		
并行从动端口通信 （PSP）	无		
外部存储器总线	无		
10 位模数转换模块	11 路输入通道		
复位 （和延时）	POR、 BOR、 RESET 指令、堆栈满、堆栈下溢、 MCLR 和 WDT （PWRT 和 OST）		
指令集	75 条指令，使能扩展指令集后总共为 83 条指令		
封装	64 引脚 TQFP		

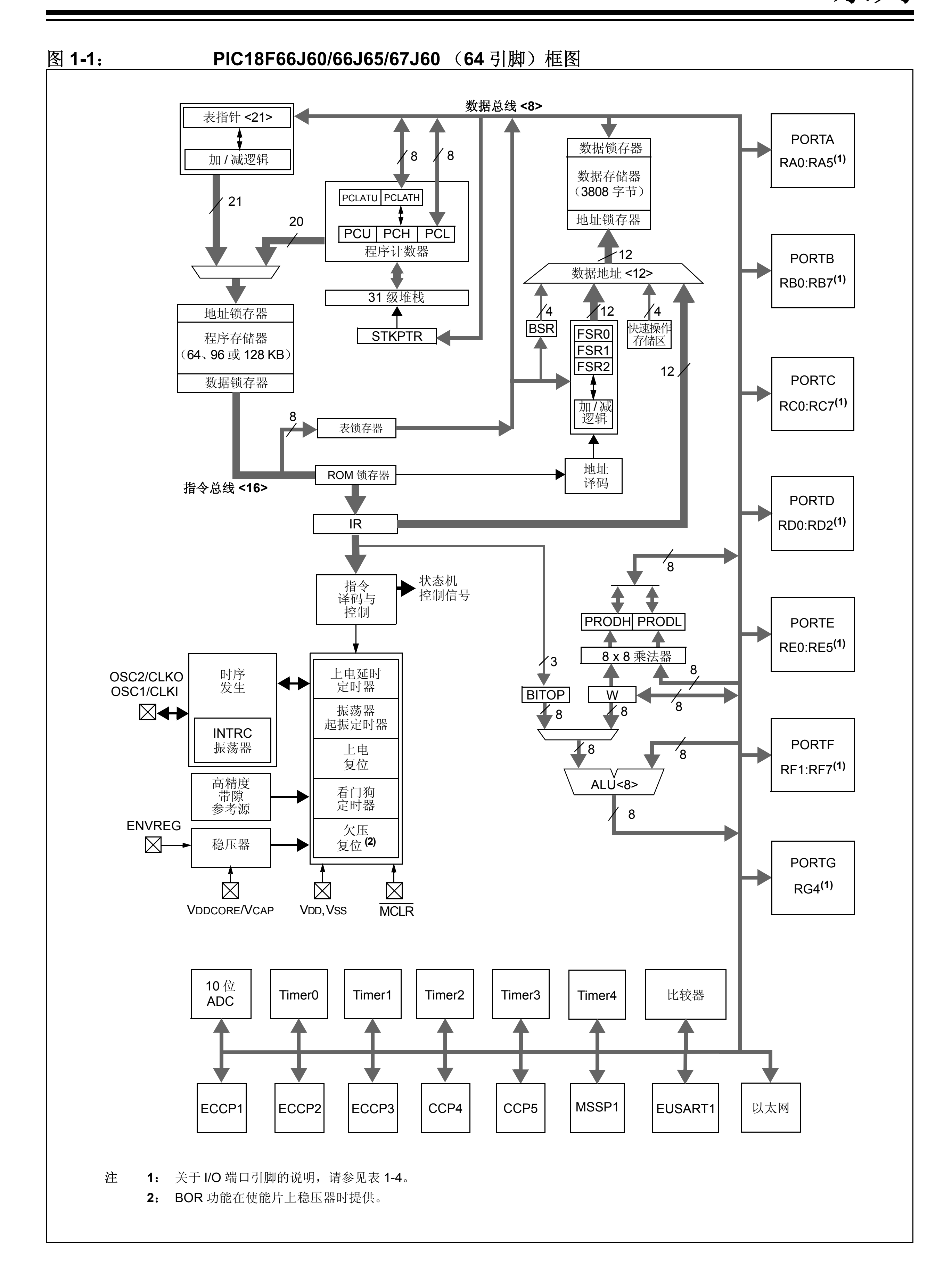
表 1-2： PIC18F97J60 系列器件特性 （80 引脚器件）

特性	PIC18F86J60	PIC18F86J65	PIC18F87J60
工作频率	DC – 41.67 MHz	DC – 41.67 MHz	DC – 41.67 MHz
程序存储器 （字节数）	64K	96K	128K
程序存储器 （指令数）	32764	49148	65532
数据存储器 （字节数）	3808		
中断源	27		
I/O 端口	端口 A, B, C, D, E, F, G, H, J		
I/O 引脚	55		
定时器	5		
捕捉 / 比较 /PWM 模块	2		
增强型捕捉 / 比较 /PWM 模块	3		
串行通信	MSSP (1) 和增强型 USART (2)		
以太网通信 （10Base-T）	有		
并行从动端口通信 （PSP）	无		
外部存储器总线	无		
10 位模数转换模块	15 路输入通道		
复位 （和延时）	POR、 BOR、 RESET 指令、堆栈满、堆栈下溢、 MCLR 和 WDT （PWRT 和 OST）		
指令集	75 条指令，使能扩展指令集后总共为 83 条指令		
封装	80 引脚 TQFP		

表 1-3： PIC18F97J60 系列器件特性 （100 引脚器件）

特性	PIC18F96J65	PIC18F97J60	PIC18F86J10
工作频率	DC – 41.67 MHz	DC – 41.67 MHz	DC – 41.67 MHz
程序存储器 （字节数）	64K	96K	128K
程序存储器 （指令数）	32764	49148	65532
数据存储器 （字节数）	3808		
中断源	29		
I/O 端口	端口 A, B, C, D, E, F, G, H, J		
I/O 引脚	70		
定时器	5		
捕捉 / 比较 /PWM 模块	2		
增强型捕捉 / 比较 /PWM 模块	3		
串行通信	MSSP (2) 和增强型 USART (2)		
以太网通信 （10Base-T）	有		
并行从动端口通信 （PSP）	有		
外部存储器总线	有		
10 位模数转换模块	16 路输入通道		
复位 （和延时）	POR、 BOR、 RESET 指令、堆栈满、堆栈下溢、 $\overline{MCLR}$ 和 WDT （PWRT 和 OST）		
指令集	75 条指令，使能扩展指令集后总共为 83 条指令		
封装	100 引脚 TQFP		

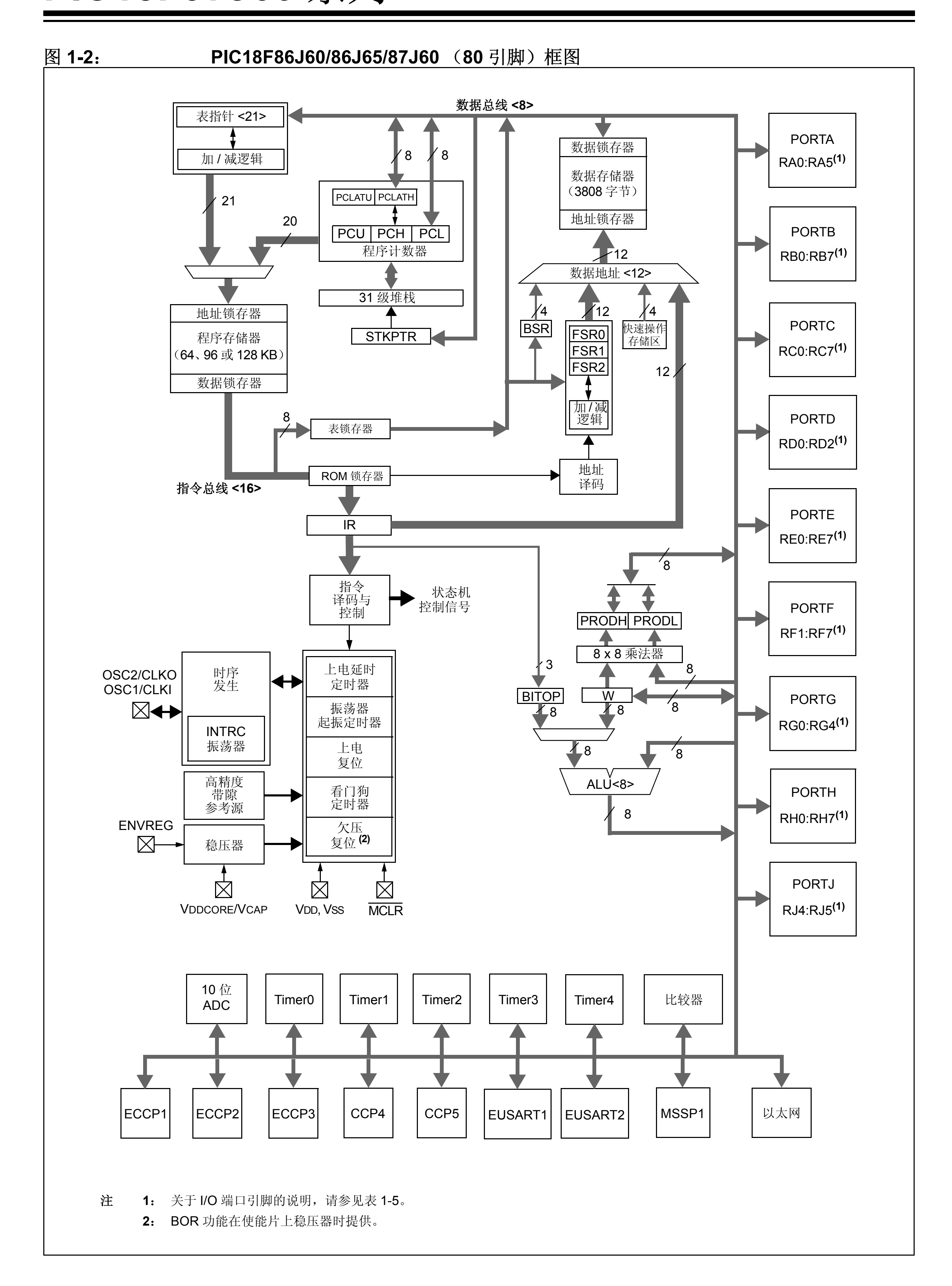
图 1-1： PIC18F66J60/66J65/67J60 （64 引脚）框图



注 1： 关于 I/O 端口引脚的说明，请参见表 1-4。

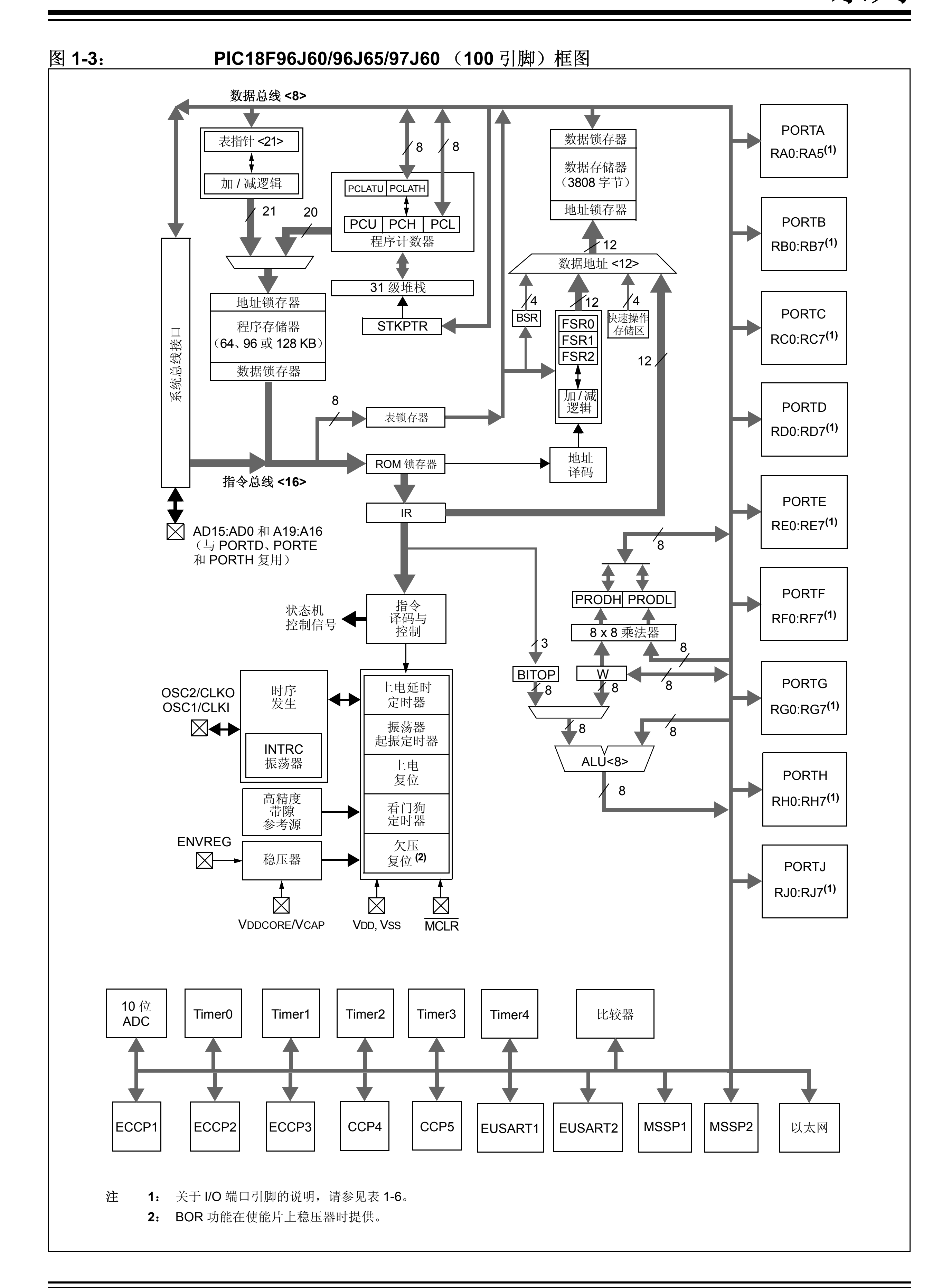
2： BOR 功能在使能片上稳压器时提供。

图 1-2： PIC18F86J60/86J65/87J60 （80 引脚）框图



注 1： 关于 I/O 端口引脚的说明，请参见表 1-5。
2： BOR 功能在使能片上稳压器时提供。

图 1-3： PIC18F96J60/96J65/97J60 （100 引脚）框图



注 1： 关于 I/O 端口引脚的说明，请参见表 1-6。
2： BOR 功能在使能片上稳压器时提供。

表 1-4：　PIC18F66J60/66J65/67J60 I/O 引脚说明

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
$\overline{MCLR}$	7	I	ST	主清零 （复位）输入。此引脚为低电平时，器件复位。
OSC1/CLKI	39			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振或外部时钟源输入。在内部 RC 模式下带 ST 缓冲区，否则带 CMOS 缓冲区。
CLKI		I	CMOS	外部时钟源输入。总是与引脚功能 OSC1 复用。（见相关的 OSC2/CLKO 引脚信息。）
OSC2/CLKO	40			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振振荡器模式下，该引脚与晶振或谐振器相连。
CLKO		O	—	在内部 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4 分频，该频率等于指令周期的倒数。
				PORTA 是双向 I/O 端口。
RA0/LEDA/AN0	24			
RA0		I/O	TTL	数字 I/O。
LEDA		O	—	以太网 LEDA 指示器输出。
AN0		I	模拟	模拟输入 0。
RA1/LEDB/AN1	23			
RA1		I/O	TTL	数字 I/O。
LEDB		O	—	以太网 LEDB 指示器输出。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-	22			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压 （低电平）输入。
RA3/AN3/VREF+	21			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压 （高电平）输入。
RA4/T0CKI	28			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
RA5/AN4	27			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

超前信息

表 1-4： PIC18F66J60/66J65/67J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTB 是双向 I/O 端口。 PORTB 在所有输入端都可软件编程为内部弱上拉。
RB0/INT0/FLT0	3			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
FLT0		I	ST	增强型 PWM 故障输入 （ECCP 模块）；通过软件使能。
RB1/INT1	4			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
RB2/INT2	5			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
RB3/INT3	6			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
RB4/KBI0	44			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1	43			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
RB6/KBI2/PGC	42			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	37			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

表 1-4： PIC18F66J60/66J65/67J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI	30			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/ECCP2/P2A	29			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
ECCP2		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
P2A		O	—	ECCP2 PWM 输出 A。
RC2/ECCP1/P1A	33			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM 1 输出。
P1A		O	—	ECCP1 PWM 输出 A。
RC3/SCK1/SCL1	34			
RC3		I/O	ST	数字 I/O。
SCK1		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL1		I/O	ST	I^2C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI1/SDA1	35			
RC4		I/O	ST	数字 I/O。
SDI1		I	ST	SPI 数据输入。
SDA1		I/O	ST	I^2C 数据 I/O。
RC5/SDO1	36			
RC5		I/O	ST	数字 I/O。
SDO1		O	—	SPI 数据输出。
RC6/TX1/CK1	31			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	32			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （见相关的 TX1/CK1 引脚信息）。

图注：
TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

表 1-4： PIC18F66J60/66J65/67J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTD 是双向 I/O 端口。
RD0/P1B	60			
RD0		I/O	ST	数字 I/O。
P1B		O	—	ECCP1 PWM 输出 B。
RD1/ECCP3/P3A	59			
RD1		I/O	ST	数字 I/O。
ECCP3		I/O	ST	捕捉 3 输入 / 比较 3 输出 /PWM 3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RD2/CCP4/P3D	58			
RD2		I/O	ST	数字 I/O。
CCP4		I/O	ST	捕捉 4 输入 / 比较 4 输出 /PWM 4 输出。
P3D		O	—	ECCP4 PWM 输出 D。

图注：
TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （没有 P 二极管接到 VDD）

表 1-4： PIC18F66J60/66J65/67J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTE 是双向 I/O 端口。
RE0/P2D	2			
RE0		I/O	ST	数字 I/O。
P2D		O	—	ECCP2 PWM 输出 D。
RE1/P2C	1			
RE1		I/O	ST	数字 I/O。
P2C		O	—	ECCP2 PWM 输出 C。
RE2/P2B	64			
RE2		I/O	ST	数字 I/O。
P2B		O	—	ECCP2 PWM 输出 B。
RE3/P3C	63			
RE3		I/O	ST	数字 I/O。
P3C		O	—	ECCP3 PWM 输出 C。
RE4/P3B	62			
RE4		I/O	ST	数字 I/O。
P3B		O	—	ECCP3 PWM 输出 B。
RE5/P1C	61			
RE5		I/O	ST	数字 I/O。
P1C		O	—	ECCP1 PWM 输出 C。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

表 **1-4**： **PIC18F66J60/66J65/67J60 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTF 是双向 I/O 端口。
RF1/AN6/C2OUT	17			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	比较器 2 的输出。
RF2/AN7/C1OUT	16			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	比较器 1 的输出。
RF3/AN8	15			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
RF4/AN9	14			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
RF5/AN10/CVREF	13			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	—	比较器参考电压输出。
RF6/AN11	12			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
RF7/$\overline{SS1}$	11			
RF7		I/O	ST	数字 I/O。
$\overline{SS1}$		I	TTL	SPI 从动选择输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路（没有 P 二极管接到 VDD）

表 1-4： PIC18F66J60/66J65/67J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTG 是双向 I/O 端口。
RG4/CCP5/P1D	8			
RG4		I/O	ST	数字 I/O。
CCP5		I/O	ST	捕捉 5 输入 / 比较 5 输出 /PWM 5 输出。
P1D		O	—	ECCP1 PWM 输出 D。
VSS	9, 25, 41, 56	P	—	逻辑和 I/O 引脚的参考地。
VDD	26, 38, 57	P	—	外设数字逻辑和 I/O 引脚的正电源。
AVSS	20	P	—	模拟模块的参考地。
AVDD	19	P	—	模拟模块的正电源。
ENVREG	18	I	ST	片上稳压器使能。
VDDCORE/VCAP	10			内核逻辑电源或外部滤波电容连接。
VDDCORE		P	—	单片机内核逻辑的正电源 （稳压器禁止）。
VCAP		P	—	外部滤波电容连接 （稳压器使能）。
VSSPLL	55	P	—	以太网 PHY PLL 的参考地。
VDDPLL	54	P	—	以太网 PHY PLL 的 3.3V 正电源。
VSSTX	52	P	—	以太网 PHY 发送子系统的参考地。
VDDTX	49	P	—	以太网 PHY 发送子系统的 3.3V 正电源。
VSSRX	45	P	—	以太网 PHY 接收子系统的参考地。
VDDRX	48	P	—	以太网 PHY 接收子系统的 3.3V 正电源。
RBIAS	53	P	—	以太网 PHY 的偏置电流。必须通过电阻连接到 VSS，有关规范请参见**第 18.0 节 “以太网模块”**。
TPOUT+	51	O	—	以太网差分信号输出。
TPOUT-	50	O	—	以太网差分信号输出。
TPIN+	47	I	模拟	以太网差分信号输入。
TPIN-	46	I	模拟	以太网差分信号输入。

图注：
TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （没有 P 二极管接到 VDD）

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
$\overline{MCLR}$	9	I	ST	主清零 （复位）输入。此引脚为低电平时，器件复位。
OSC1/CLKI	49			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振或外部时钟源输入。在内部 RC 模式下带 ST 缓冲区，否则带 CMOS 缓冲区。
CLKI		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能复用。（见相关的 OSC2/CLKO 引脚信息。）
OSC2/CLKO	50			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振振荡器模式下，该引脚与晶振或谐振器相连。
CLKO		O	—	在内部 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4 分频，该频率等于指令周期的倒数。
				PORTA 是双向 I/O 端口。
RA0/LEDA/AN0	30			
RA0		I/O	TTL	数字 I/O。
LEDA		O	—	以太网 LEDA 指示器输出。
AN0		I	模拟	模拟输入 0。
RA1/LEDB/AN1	29			
RA1		I/O	TTL	数字 I/O。
LEDB		O	—	以太网 LEDB 指示器输出。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-	28			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压 （低电平）输入。
RA3/AN3/VREF+	27			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压 （高电平）输入。
RA4/T0CKI	34			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
RA5/AN4	33			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTB 是双向 I/O 端口。 PORTB 在所有输入端都可软件编程为内部弱上拉。
RB0/INT0/FLT0	5			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
FLT0		I	ST	增强型 PWM 故障输入 （ECCP 模块）；通过软件使能。
RB1/INT1	6			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
RB2/INT2	7			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
RB3/INT3	8			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
RB4/KBI0	54			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1	53			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
RB6/KBI2/PGC	52			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	47			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

注 1： CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI	36			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/ECCP2/P2A	35			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
ECCP2[(1)]		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
P2A[(1)]		O	—	ECCP2 PWM 输出 A。
RC2/ECCP1/P1A	43			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM 1 输出。
P1A		O	—	ECCP1 PWM 输出 A。
RC3/SCK1/SCL1	44			
RC3		I/O	ST	数字 I/O。
SCK1		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL1		I/O	ST	I²C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI1/SDA1	45			
RC4		I/O	ST	数字 I/O。
SDI1		I	ST	SPI 数据输入。
SDA1		I/O	ST	I²C 数据 I/O。
RC5/SDO1	46			
RC5		I/O	ST	数字 I/O。
SDO1		O	—	SPI 数据输出。
RC6/TX1/CK1	37			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	38			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （见相关的 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTD 是双向 I/O 端口。
RD0	72	I/O	ST	数字 I/O。
RD1	69	I/O	ST	数字 I/O。
RD2	68	I/O	ST	数字 I/O。
				PORTE 是双向 I/O 端口。
RE0/P2D	4			
RE0		I/O	ST	数字 I/O。
P2D		O	—	ECCP2 PWM 输出 D。
RE1/P2C	3			
RE1		I/O	ST	数字 I/O。
P2C		O	—	ECCP2 PWM 输出 C。
RE2/P2B	78			
RE2		I/O	ST	数字 I/O。
P2B		O	—	ECCP2 PWM 输出 B。
RE3/P3C	77			
RE3		I/O	ST	数字 I/O。
P3C(2)		O	—	ECCP3 PWM 输出 C。
RE4/P3B	76			
RE4		I/O	ST	数字 I/O。
P3B(2)		O	—	ECCP3 PWM 输出 B。
RE5/P1C	75			
RE5		I/O	ST	数字 I/O。
P1C(2)		O	—	ECCP1 PWM 输出 C。
RE6/P1B	74			
RE6		I/O	ST	数字 I/O。
P1B(2)		O	—	ECCP1 PWM 输出 B。
RE7/ECCP2/P2A	73			
RE7		I/O	ST	数字 I/O。
ECCP2(3)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
P2A(3)		O	—	ECCP2 PWM 输出 A。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTF 是双向 I/O 端口。
RF1/AN6/C2OUT	23			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	比较器 2 的输出。
RF2/AN7/C1OUT	18			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	比较器 1 的输出。
RF3/AN8	17			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
RF4/AN9	16			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
RF5/AN10/CVREF	15			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	—	比较器参考电压输出。
RF6/AN11	14			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
RF7/$\overline{SS1}$	13			
RF7		I/O	ST	数字 I/O。
$\overline{SS1}$		I	TTL	SPI 从动选择输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTG 是双向 I/O 端口。
RG0/ECCP3/P3A	56			
RG0		I/O	ST	数字 I/O。
ECCP3		I/O	ST	捕捉 3 输入 / 比较 3 输出 /PWM 3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RG1/TX2/CK2	55			
RG1		I/O	ST	数字 I/O。
TX2		O	—	EUSART2 异步发送。
CK2		I/O	ST	EUSART2 同步时钟 （见相关的 RX2/DT2 引脚信息）。
RG2/RX2/DT2	42			
RG2		I/O	ST	数字 I/O。
RX2		I	ST	EUSART2 异步接收。
DT2		I/O	ST	EUSART2 同步数据 （见相关的 TX2/CK2 引脚信息）。
RG3/CCP4/P3D	41			
RG3		I/O	ST	数字 I/O。
CCP4		I/O	ST	捕捉 4 输入 / 比较 4 输出 /PWM 4 输出。
P3D		O	—	ECCP3 PWM 输出 D。
RG4/CCP5/P1D	10			
RG4		I/O	ST	数字 I/O。
CCP5		I/O	ST	捕捉 5 输入 / 比较 5 输出 /PWM 5 输出。
P1D		O	—	ECCP1 PWM 输出 D。

图注： TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 二极管接到 VDD）

注 **1：** CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTH 是双向 I/O 端口。
RH0	79	I/O	ST	数字 I/O。
RH1	80	I/O	ST	数字 I/O。
RH2	1	I/O	ST	数字 I/O。
RH3	2	I/O	ST	数字 I/O。
RH4/AN12/P3C	22			
RH4		I/O	ST	数字 I/O。
AN12		I	模拟	模拟输入 12。
P3C(4)		O	—	ECCP3 PWM 输出 C。
RH5/AN13/P3B	21			
RH5		I/O	ST	数字 I/O。
AN13		I	模拟	模拟输入 13。
P3B(4)		O	—	ECCP3 PWM 输出 B。
RH6/AN14/P1C	20			
RH6		I/O	ST	数字 I/O。
AN14		I	模拟	模拟输入 14。
P1C(4)		O	—	ECCP1 PWM 输出 C。
RH7/AN15/P1B	19			
RH7		I/O	ST	数字 I/O。
AN15		I	模拟	模拟输入 15。
P1B(4)		O	—	ECCP1 PWM 输出 B。

图注：
TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

注 1： CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-5： PIC18F86J60/86J65/87J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTJ 是双向 I/O 端口。
RJ4	39	I/O	ST	数字 I/O。
RJ5	40	I/O	ST	数字 I/O
VSS	11, 31, 51, 70	P	—	逻辑和 I/O 引脚的参考地。
VDD	32, 48, 71	P	—	外设数字逻辑和 I/O 引脚的正电源。
AVSS	26	P	—	模拟模块的参考地。
AVDD	25	P	—	模拟模块的正电源。
ENVREG	24	I	ST	片上稳压器使能。
VDDCORE/VCAP	12			内核逻辑电源或外部滤波电容连接。
VDDCORE		P	—	单片机内核逻辑的正电源 （稳压器禁止）。
VCAP		P	—	外部滤波电容连接 （稳压器使能）。
VSSPLL	67	P	—	以太网 PHY PLL 的参考地。
VDDPLL	66	P	—	以太网 PHY PLL 的 3.3V 正电源。
VSSTX	64	P	—	以太网 PHY 发送子系统的参考地。
VDDTX	61	P	—	以太网 PHY 发送子系统的 3.3V 正电源。
VSSRX	57	P	—	以太网 PHY 接收子系统的参考地。
VDDRX	60	P	—	以太网 PHY 接收子系统的 3.3V 正电源。
RBIAS	65	P	—	以太网 PHY 的偏置电流。必须通过电阻连接到 VSS，有关规范请参见**第 18.0 节 "以太网模块"**。
TPOUT+	63	O	—	以太网差分信号输出。
TPOUT-	62	O	—	以太网差分信号输出。
TPIN+	59	I	模拟	以太网差分信号输入。
TPIN-	58	I	模拟	以太网差分信号输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 **1：** CCP2MX 配置位置 1 时， ECCP2/P2A 的默认配置。
2： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
3： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置。
4： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

超前信息

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
$\overline{\text{MCLR}}$	13	I	ST	主清零 （复位）输入。此引脚为低电平时，器件复位。
OSC1/CLKI	63			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振或外部时钟源输入。在内部 RC 模式下带 ST 缓冲区，否则带 CMOS 缓冲区。
CLKI		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能复用。（见相关的 OSC2/CLKO 引脚信息。）
OSC2/CLKO	64			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振振荡器模式下，该引脚与晶振或谐振器相连。
CLKO		O	—	在内部 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4分频，该频率等于指令周期的倒数。
				PORTA 是双向 I/O 端口。
RA0/LEDA/AN0	35			
RA0		I/O	TTL	数字 I/O。
LEDA		O	—	以太网 LEDA 指示器输出。
AN0		I	模拟	模拟输入 0。
RA1/LEDB/AN1	34			
RA1		I/O	TTL	数字 I/O。
LEDB		O	—	以太网 LEDB 指示器输出。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-	33			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压 （低电平）输入。
RA3/AN3/VREF+	32			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压 （高电平）输入。
RA4/T0CKI	42			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
RA5/AN4	41			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 **1：** CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTB 是双向 I/O 端口。 PORTB 在所有输入端都可软件编程为内部弱上拉。
RB0/INT0/FLT0	5			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
FLT0		I	ST	增强型 PWM 故障输入 （ECCP 模块）；通过软件使能。
RB1/INT1	6			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
RB2/INT2	7			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
RB3/INT3/ECCP2/P2A	8			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
ECCP2[(1)]		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
P2A[(1)]		O	—	ECCP2 PWM 输出 A。
RB4/KBI0	69			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1	68			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
RB6/KBI2/PGC	67			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	57			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI	44			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/ECCP2/P2A	43			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
ECCP2(2)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
P2A(2)		O	—	ECCP2 PWM 输出 A。
RC2/ECCP1/P1A	53			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM 1 输出。
P1A		O	—	ECCP1 PWM 输出 A。
RC3/SCK1/SCL1	54			
RC3		I/O	ST	数字 I/O。
SCK1		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL1		I/O	ST	I^{2}C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI1/SDA1	55			
RC4		I/O	ST	数字 I/O。
SDI1		I	ST	SPI 数据输入。
SDA1		I/O	ST	I^{2}C 数据 I/O。
RC5/SDO1	56			
RC5		I/O	ST	数字 I/O。
SDO1		O	—	SPI 数据输出。
RC6/TX1/CK1	45			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	46			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （见相关的 TX1/CK1 引脚信息）。

图注：
TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTD 是双向 I/O 端口。
RD0/AD0/PSP0	92			
RD0		I/O	ST	数字 I/O。
AD0		I/O	TTL	外部存储器地址 / 数据 0。
PSP0		I/O	TTL	并行从动端口数据。
RD1/AD1/PSP1	91			
RD1		I/O	ST	数字 I/O。
AD1		I/O	TTL	外部存储器地址 / 数据 1。
PSP1		I/O	TTL	并行从动端口数据。
RD2/AD2/PSP2	90			
RD2		I/O	ST	数字 I/O。
AD2		I/O	TTL	外部存储器地址 / 数据 2。
PSP2		I/O	TTL	并行从动端口数据。
RD3/AD3/PSP3	89			
RD3		I/O	ST	数字 I/O。
AD3		I/O	TTL	外部存储器地址 / 数据 3。
PSP3		I/O	TTL	并行从动端口数据。
RD4/AD4/PSP4/SDO2	88			
RD4		I/O	ST	数字 I/O。
AD4		I/O	TTL	外部存储器地址 / 数据 4。
PSP4		I/O	TTL	并行从动端口数据。
SDO2		O	—	SPI 数据输出。
RD5/AD5/PSP5/SDI2/SDA2	87			
RD5		I/O	ST	数字 I/O。
AD5		I/O	TTL	外部存储器地址 / 数据 5。
PSP5		I/O	TTL	并行从动端口数据。
SDI2		I	ST	SPI 数据输入。
SDA2		I/O	ST	I^2C™ 数据 I/O。
RD6/AD6/PSP6/SCK2/SCL2	84			
RD6		I/O	ST	数字 I/O。
AD6		I/O	TTL	外部存储器地址 / 数据 6。
PSP6		I/O	TTL	并行从动端口数据。
SCK2		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL2		I/O	ST	I^2C 模式的同步串行时钟输入 / 输出。
RD7/AD7/PSP7/$\overline{SS2}$	83			
RD7		I/O	ST	数字 I/O。
AD7		I/O	TTL	外部存储器地址 / 数据 7。
PSP7		I/O	TTL	并行从动端口数据。
$\overline{SS2}$		I	TTL	SPI 从动选择输入。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 **1-6：** **PIC18F96J60/96J65/97J60 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTE 是双向 I/O 端口。
RE0/AD8/$\overline{RD}$/P2D	4			
RE0		I/O	ST	数字 I/O。
AD8		I/O	TTL	外部存储器地址 / 数据 8。
$\overline{RD}$		I	TTL	并行从动端口的读控制。
P2D		O	—	ECCP2 PWM 输出 D。
RE1/AD9/$\overline{WR}$/P2C	3			
RE1		I/O	ST	数字 I/O。
AD9		I/O	TTL	外部存储器地址 / 数据 9。
$\overline{WR}$		I	TTL	并行从动端口的写控制。
P2C		O	—	ECCP2 PWM 输出 C。
RE2/AD10/$\overline{CS}$/P2B	98			
RE2		I/O	ST	数字 I/O。
AD10		I/O	TTL	外部存储器地址 / 数据 10。
$\overline{CS}$		I	TTL	并行从动端口的片选控制。
P2B		O	—	ECCP2 PWM 输出 B。
RE3/AD11/P3C	97			
RE3		I/O	ST	数字 I/O。
AD11		I/O	TTL	外部存储器地址 / 数据 11。
P3C**(3)**		O	—	ECCP3 PWM 输出 C。
RE4/AD12/P3B	96			
RE4		I/O	ST	数字 I/O。
AD12		I/O	TTL	外部存储器地址 / 数据 12。
P3B**(3)**		O	—	ECCP3 PWM 输出 B。
RE5/AD13/P1C	95			
RE5		I/O	ST	数字 I/O。
AD13		I/O	TTL	外部存储器地址 / 数据 13。
P1C**(3)**		O	—	ECCP1 PWM 输出 C。
RE6/AD14/P1B	94			
RE6		I/O	ST	数字 I/O。
AD14		I/O	TTL	外部存储器地址 / 数据 14。
P1B**(3)**		O	—	ECCP1 PWM 输出 B。
RE7/AD15/ECCP2/P2A	93			
RE7		I/O	ST	数字 I/O。
AD15		I/O	TTL	外部存储器地址 / 数据 15。
ECCP2**(4)**		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
P2A**(4)**		O	—	ECCP2 PWM 输出 A。

图注： TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

注 **1：** CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTF 是双向 I/O 端口。
RF0/AN5	12			
RF0		I/O	ST	数字 I/O。
AN5		I	模拟	模拟输入 5。
RF1/AN6/C2OUT	28			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	比较器 2 的输出。
RF2/AN7/C1OUT	23			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	比较器 1 的输出。
RF3/AN8	22			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
RF4/AN9	21			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
RF5/AN10/CVREF	20			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	—	比较器参考电压输出。
RF6/AN11	19			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
RF7/$\overline{\text{SS1}}$	18			
RF7		I/O	ST	数字 I/O。
$\overline{\text{SS1}}$		I	TTL	SPI 从动选择输入。

图注：
TTL = TTL 兼容输入
CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器
Analog = 模拟输入
I = 输入
O = 输出
P = 电源
OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTG 是双向 I/O 端口。
RG0/ECCP3/P3A	71			
RG0		I/O	ST	数字 I/O。
ECCP3		I/O	ST	捕捉 3 输入 / 比较 3 输出 /PWM 3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RG1/TX2/CK2	70			
RG1		I/O	ST	数字 I/O。
TX2		O	—	EUSART2 异步发送。
CK2		I/O	ST	EUSART2 同步时钟 （见相关的 RX2/DT2 引脚信息）。
RG2/RX2/DT2	52			
RG2		I/O	ST	数字 I/O。
RX2		I	ST	EUSART2 异步接收。
DT2		I/O	ST	EUSART2 同步数据 （见相关的 TX2/CK2 引脚信息）。
RG3/CCP4/P3D	51			
RG3		I/O	ST	数字 I/O。
CCP4		I/O	ST	捕捉 4 输入 / 比较 4 输出 /PWM 4 输出。
P3D		O	—	ECCP3 PWM 输出 D。
RG4/CCP5/P1D	14			
RG4		I/O	ST	数字 I/O。
CCP5		I/O	ST	捕捉 5 输入 / 比较 5 输出 /PWM 5 输出。
P1D		O	—	ECCP1 PWM 输出 D。
RG5	11	I/O	ST	数字 I/O。
RG6	10	I/O	ST	数字 I/O。
RG7	38	I/O	ST	数字 I/O。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTH 是双向 I/O 端口。
RH0/A16	99			
RH0		I/O	ST	数字 I/O。
A16		O	—	外部存储器地址 16。
RH1/A17	100			
RH1		I/O	ST	数字 I/O。
A17		O	—	外部存储器地址 17。
RH2/A18	1			
RH2		I/O	ST	数字 I/O。
A18		O	—	外部存储器地址 18。
RH3/A19	2			
RH3		I/O	ST	数字 I/O。
A19		O	—	外部存储器地址 19。
RH4/AN12/P3C	27			
RH4		I/O	ST	数字 I/O。
AN12		I	模拟	模拟输入 12。
P3C[5]		O	—	ECCP3 PWM 输出 C。
RH5/AN13/P3B	26			
RH5		I/O	ST	数字 I/O。
AN13		I	模拟	模拟输入 13。
P3B[5]		O	—	ECCP3 PWM 输出 B。
RH6/AN14/P1C	25			
RH6		I/O	ST	数字 I/O。
AN14		I	模拟	模拟输入 14。
P1C[5]		O	—	ECCP1 PWM 输出 C。
RH7/AN15/P1B	24			
RH7		I/O	ST	数字 I/O。
AN15		I	模拟	模拟输入 15。
P1B[5]		O	—	ECCP1 PWM 输出 B。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 V_{DD}）

注 **1：** CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
				PORTJ 是双向 I/O 端口。
RJ0/ALE	49			
RJ0		I/O	ST	数字 I/O。
ALE		O	—	外部存储器地址锁存器使能。
RJ1/$\overline{OE}$	50			
RJ1		I/O	ST	数字 I/O。
$\overline{OE}$		O	—	外部存储器输出使能。
RJ2/$\overline{WRL}$	66			
RJ2		I/O	ST	数字 I/O。
$\overline{WRL}$		O	—	外部存储器低字节写控制。
RJ3/$\overline{WRH}$	61			
RJ3		I/O	ST	数字 I/O。
$\overline{WRH}$		O	—	外部存储器高字节写控制。
RJ4/BA0	47			
RJ4		I/O	ST	数字 I/O。
BA0		O	—	外部存储器字节地址 0 控制。
RJ5/$\overline{CE}$	48			
RJ5		I/O	ST	数字 I/O
$\overline{CE}$		O	—	外部存储器芯片使能控制。
RJ6/$\overline{LB}$	58			
RJ6		I/O	ST	数字 I/O。
$\overline{LB}$		O	—	外部存储器低字节控制。
RJ7/$\overline{UB}$	39			
RJ7		I/O	ST	数字 I/O。
$\overline{UB}$		O	—	外部存储器高字节控制。

图注： TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

表 1-6： PIC18F96J60/96J65/97J60 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲区类型	说明
NC	9	—	—	无连接。
VSS	15, 36, 40, 60, 65, 85	P	—	逻辑和 I/O 引脚的参考地。
VDD	17, 37, 59, 62, 86	P	—	外设数字逻辑和 I/O 引脚的正电源。
AVSS	31	P	—	模拟模块的参考地。
AVDD	30	P	—	模拟模块的正电源。
ENVREG	29	I	ST	片上稳压器使能。
VDDCORE/VCAP	16			内核逻辑电源或外部滤波电容连接。
VDDCORE		P	—	单片机内核逻辑的正电源 （稳压器禁止）。
VCAP		P	—	外部滤波电容连接 （稳压器使能）。
VSSPLL	82	P	—	以太网 PHY PLL 的参考地。
VDDPLL	81	P	—	以太网 PHY PLL 的 3.3V 正电源。
VSSTX	79	P	—	以太网 PHY 发送子系统的参考地。
VDDTX	76	P	—	以太网 PHY 发送子系统的 3.3V 正电源。
VSSRX	72	P	—	以太网 PHY 接收子系统的参考地。
VDDRX	75	P	—	以太网 PHY 接收子系统的 3.3V 正电源。
RBIAS	80	P	—	以太网 PHY 的偏置电流。必须通过电阻连接到 Vss，有关规范请参见**第 18.0 节 “以太网模块”**。
TPOUT+	78	O	—	以太网差分信号输出。
TPOUT-	77	O	—	以太网差分信号输出。
TPIN+	74	I	模拟	以太网差分信号输入。
TPIN-	73	I	模拟	以太网差分信号输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 二极管接到 VDD）

注 1： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （扩展单片机模式）。
2： 在所有操作模式中，所有器件的 ECCP2/P2A 的默认配置 （CCP2MX 配置位置 1）。
3： P1B/P1C/P3B/P3C 的默认配置 （ECCPMX 配置位置 1）。
4： CCP2MX 配置位清零时， ECCP2/P2A 的替代配置 （单片机模式）。
5： P1B/P1C/P3B/P3C 的替代配置 （ECCPMX 配置位清零）。

2.0 振荡器配置

2.1 概述

PIC18F97J60 系列器件的振荡器和单片机时钟系统不同于标准 PIC18FXXJXX 器件的。新增的以太网模块需要一个稳定的 25 MHz 时钟源，因此必须为其配备能提供该频率的主振荡器以及不同范围的单片机时钟速度。图 2-1 中简要说明了振荡器的结构。

PIC18FXXJXX 增强型单片机中使用的其他振荡器功能（例如内部 RC 振荡器和时钟切换）仍保持不变。它们将在本章中稍后讨论。

2.2 振荡器类型

PIC18F97J60 系列器件可以在 5 种不同的振荡器模式下工作：

1. HS 高速晶振 / 谐振器
2. HSPLL 带软件 PLL 控制的高速晶振 / 谐振器
3. EC 带 Fosc/4 输出的外部时钟
4. ECPLL 带软件 PLL控制的外部时钟
5. INTRC 31 kHz 内部振荡器

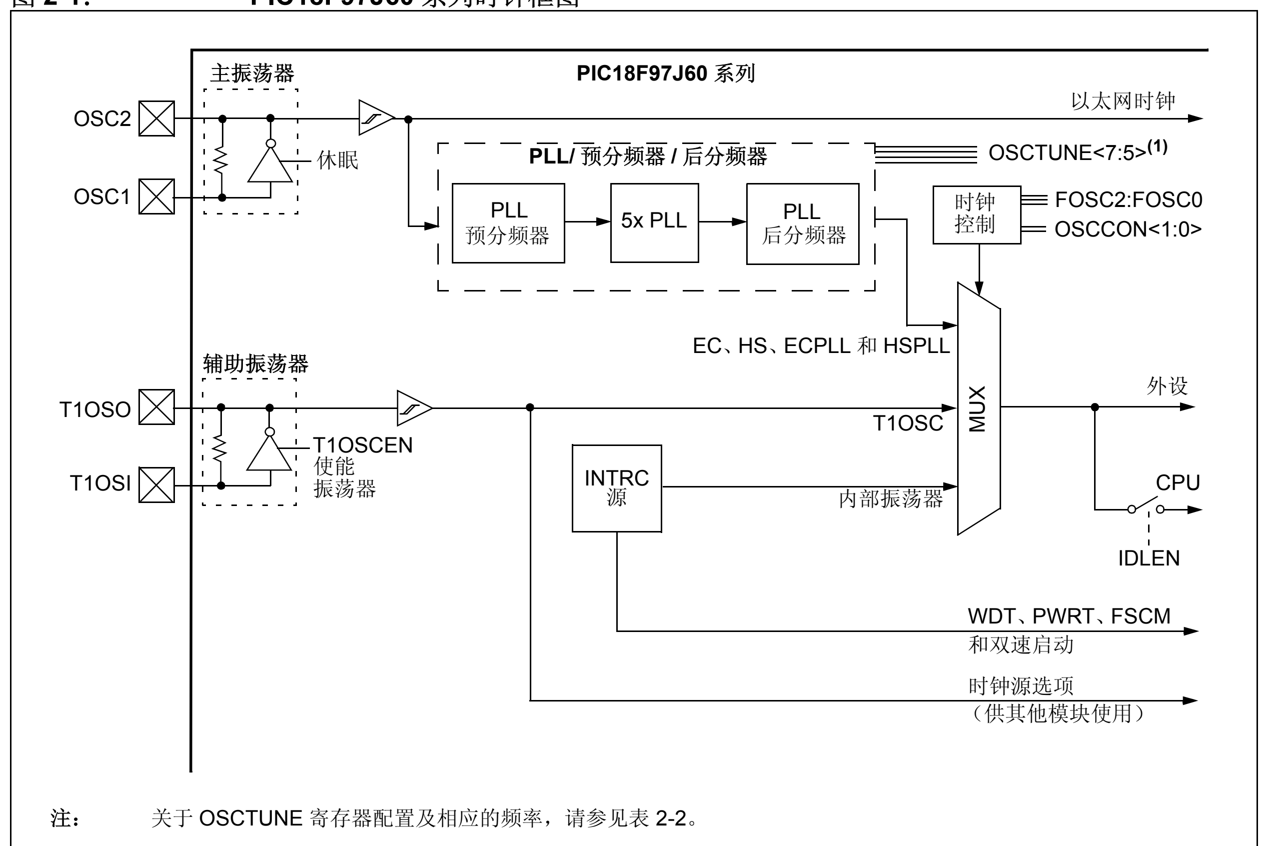
2.2.1 振荡器控制

可以通过编程 FOSC2:FOSC0 配置位来选择振荡器模式。 FOSC1:FOSC0 位选择默认的主振荡器模式，而 FOSC2 选择何时调用 INTRC。

OSCCON 寄存器 （寄存器 2-2）选择有效时钟模式。它主要用于在功耗管理模式下控制时钟切换。它的使用在**第 2.7.1 节 “振荡器控制寄存器”**中讨论。

通过选择预分频器 / 后分频器组合设置以及使能 PLL， OSCTUNE 寄存器 （寄存器 2-1）用于选择主振荡器源的系统时钟频率。它的使用在**第 2.6.1 节 “PLL 模块”**中讨论。

图 2-1： PIC18F97J60 系列时钟框图



注： 关于 OSCTUNE 寄存器配置及相应的频率，请参见表 2-2。

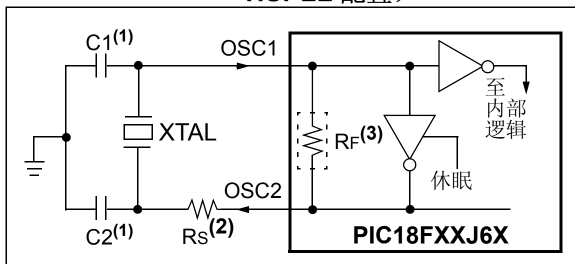
2.3 晶振 / 陶瓷谐振器 （HS 模式）

在 HS 或 HSPLL 振荡器模式下，晶振与 OSC1 和 OSC2 引脚相连来产生振荡信号。图2-2显示了引脚连接方式。

振荡器的设计要求使用平行切割的晶体。

> **注：** 使用顺序切割的晶体，可能会使振荡器产生的频率超出晶体制造厂商所给出的参数范围。

图 2-2： 晶振工作原理 （HS 或 HSPLL 配置）



注 1：关于 C1 和 C2 的初始值，请参见表 2-1。
2：对于 AT 条形切割的晶体可能会需要一个串联电阻 （RS）。
3：RF 的值随选定的振荡器模式变化。

表 2-1： 晶振的电容选择

振荡器类型	晶振频率	已测试的典型电容值：	
		C1	C2
HS	25 MHz	15 pF	15 pF

上述电容值仅供设计参考。

要得到理想的振荡器工作状况，可能需要不同的电容值。用户应当在设计的 VDD 和温度条件下测试振荡器的性能。请参见以下应用笔记以获取振荡器具体信息:

- *AN588，“PIC® Microcontroller Oscillator Design Guide”*
- *AN826，“Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices”*
- *AN849，“Basic PIC® Oscillator Design”*
- *AN943，“Practical PIC® Oscillator Analysis and Design”*
- *AN949，“Making Your Oscillator Work”*

更多信息，请参见本表下方的 “注”。

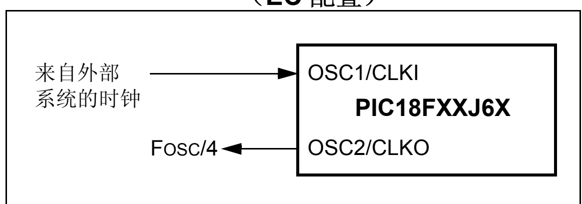
> **注** 1：电容值越大，振荡器的稳定性越高，但同时起振时间也越长。
> 2：因为每种晶振都有其自身特性，用户应当向晶振制造厂商询问外部元件的适当值。
> 3：可能需要使用电阻Rs以避免对低驱动规格的晶体造成过驱动。
> 4：请始终在设计的 VDD 和温度范围下验证振荡器性能。

2.4 外部时钟输入 （EC 模式）

EC 和 ECPLL 振荡器模式要求 OSC1 引脚与一个外部时钟源相连。在上电复位后或从休眠模式退出后，不需要振荡器起振时间。

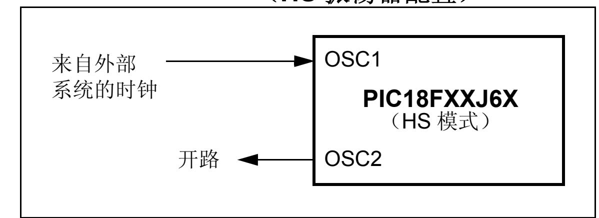
在 EC 振荡器模式下，由 OSC2 引脚输出振荡器频率的 4 分频信号。此信号可用于测试或同步其他逻辑。图 2-3 显示了 EC 振荡器模式的引脚连接方式。

图 2-3： 外部时钟输入工作原理 （EC 配置）



如图 2-4 所示，在 HS 模式下， OSC1 引脚也可以连接外部时钟源。在这种配置下，OSC2引脚保持开路状态。

图 2-4： 外部时钟输入工作原理 （HS 振荡器配置）



 超前信息

2.5 内部振荡器模块

PIC18F97J60 系列器件包含一个内部振荡器源（INTRC），它提供了标称值为 31 kHz 的输出。INTRC 在器件上电时被使能，并在器件的配置阶段为其提供时钟直到器件进入工作模式为止。如果选择 INTRC 作为器件的时钟源或者使能以下任一功能时，也会使能 INTRC：

- 故障保护时钟监视器
- 看门狗定时器
- 双速启动

在**第 24.0 节 "CPU 的特殊功能"**中详细讨论了这些功能。

也可以通过将 FOSC2 配置位置 1，将 INTRC 配置为器件启动时的默认时钟源。这将在**第 2.7.1 节 "振荡器控制寄存器"**中讨论。

2.6 以太网操作和单片机时钟

尽管PIC18F97J60系列器件可接受范围很广的晶振和外部振荡器输入信号，但它们用在以太网应用中时，必须始终使用 25 MHz 时钟源。不支持从其他频率的主振荡器源内部生成所需的以太网时钟。频率的误差范围已规定，基本上排除了使用陶瓷谐振器的可能性。更多详细信息，请参见**第 27.0 节 "电气特性"**的表 27-6 中的参数 5。

2.6.1 PLL 模块

为了适应各种应用和单片机时钟速度，在时钟系统中包括了独立的 PLL 模块。它由 3 个元件组成：

- 可配置的预分频器 （1:2 或 1:3）
- 5x PLL 倍频器
- 可配置的后分频器 （1:1、 1:2 或 1:3）

PLL 模块的工作受 OSCTUNE 寄存器（寄存器 2-1）的控制。PLL 模块的预分频器和后分频器可以和 PLL 自身一起使用或独立使用，提供了一定范围的可供选择的系统时钟频率，包括不能改变的主振荡器频率 25 MHz。表 2-2 中给出了与操作兼容的所有可能的振荡器配置。

寄存器 2-1： OSCTUNE：PLL 模块控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
PPST1	PLLEN(1)	PPST0	PPRE	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **PPST1**：PLL 后分频器配置位
1 = 2 分频
0 = 3 分频

bit 6 **PLLEN**：5x 倍频器 PLL 使能位 (1)
1 = 使能 PLL
0 = 禁止 PLL

bit 5 **PPST0**：PLL 后分频器使能位
1 = 使能后分频器
0 = 禁止后分频器

bit 4 **PPRE**：PLL 预分频器配置位
1 = 2 分频
0 = 3 分频

bit 3-0 未用：读为 0

注 1： 仅在 ECPLL 和 HSPLL 振荡器配置中可用；其他情况下，此位不可用，并且读为 0。

表 2-2： 不同 PLL 模块配置的器件时钟速度

5x PLL	PLL 预分频器	PLL 后分频器	PLL 模块配置（OSCTUNE<7:4>）	时钟频率 （MHz）
使能	÷2	禁止	x101	（注 1）
		÷2	1111	31.2500
		÷3	0111	20.8333
	÷3	禁止	x100	41.6667
		÷2	1110	20.8333
		÷3	0110	13.8899
禁止	禁止 (2)	禁止	x00x	25 （默认值）
	÷2	÷2	1011	6.2500
		÷3	0010	4.1667
	÷3	÷2	1011	4.1667
		÷3	1000	2.7778

图注： x = 无关位

注 1： 未实现的配置；代表超出单片机运行范围的时钟频率。

2： PLL 和后分频器都禁止时，预分频器自动禁止。

2.7 时钟源与振荡器切换

PIC18F97J60 系列器件包含了允许器件时钟源从主振荡器切换到其他时钟源的功能。这些器件还提供 2 个备用时钟源。当使能备用时钟源时，可以使用多种功耗管理工作模式。

基本上，这些器件都有 3 种时钟源：

- 主振荡器
- 辅助振荡器
- 内部振荡器模块

主振荡器包括外部晶振、谐振器模式和外部时钟模式。特定的模式由 FOSC2:FOSC0 配置位定义。这些模式的详细信息已在本章前面的内容中作过介绍。

辅助振荡器是指那些不与 OSC1 或 OSC2 引脚连接的外部时钟源。这些时钟源即使在控制器处于功耗管理模式时仍然可以继续工作。PIC18F97J60 系列器件支持使用 Timer1 振荡器作为辅助振荡器。此振荡器在所有功耗管理模式中通常用作实时时钟等功能的时基。

大部分情况下，在 RC0/T1OSO/T13CKI 和 RC1/T1OSI 引脚之间接有一个 32.768 kHz 的时钟晶振。在每个引脚与地之间均接有负载电容。将在**第 12.3 节 “Timer1 振荡器”**中详细讨论 Timer1 振荡器。

除了作为主时钟源之外，**内部振荡器**还可以作为功耗管理模式的时钟源。INTRC 源也可作为几种特殊功能部件（例如 WDT 和故障保护时钟监视器）的时钟源。

图 2-1 显示了 PIC18F97J60 系列器件的时钟源。关于配置寄存器的详细信息，请参见**第 24.0 节 “CPU 的特殊功能”**。

2.7.1 振荡器控制寄存器

OSCCON 寄存器（寄存器 2-2）控制全功耗模式和功耗管理模式下器件时钟工作的多个方面。

系统时钟选择位 SCS1:SCS0 用于选择时钟源。可用的时钟源包括主时钟（由 FOSC2:FOSC0 配置位定义）、辅助时钟 （Timer1 振荡器）和内部振荡器。当更改一个或多个位之后，接着是一段很短的时钟转换间隔，然后时钟源会改变。

OSTS （OSCCON<3>）和 T1RUN （T1CON<6>）位指出当前提供器件时钟的是哪一个时钟源。 T1RUN 位置 1 表明 Timer1 振荡器正在辅助时钟模式下提供器件时钟。在功耗管理模式下，任何时候这些位中只有一个会置 1。如果这些位都没有置 1，则表示当前时钟源是 INTRC，或内部振荡器刚刚起振且尚未稳定。

IDLEN 位决定当执行 SLEEP 指令时器件是进入休眠模式还是某个空闲模式。

第 3.0 节“功耗管理模式”更详细地讨论了 OSCCON 寄存器中标志位和控制位的使用。

> 注 1：要选择辅助时钟源，必须使能 Timer1 振荡器。通过将 Timer1 控制寄存器中的 T1OSCEN 位 （T1CON<3>）置 1，可以使能 Timer1 振荡器。如果未使能 Timer1 振荡器，则选择辅助时钟源的任何尝试都会被忽略。
>
> 2：建议在Timer1振荡器稳定工作之后再执行 SLEEP 指令，否则当 Timer1 振荡器起振时可能会发生很长的延时。

寄存器 2-2： OSCCON：振荡器控制寄存器

R/W-0	U-0	U-0	U-0	R-q	U-0	R/W-0	R/W-0
IDLEN	—	—	—	OSTS(1)	—	SCS1	SCS0
bit 7							bit 0

图注：	q = 由配置决定的值		
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **IDLEN：**空闲使能位
1 = 执行 SLEEP 指令后器件进入空闲模式
0 = 执行 SLEEP 指令后器件进入休眠模式

bit 6-4 **未用：**读为 0

bit 3 **OSTS：**振荡器状态位 (1)
1 = 器件使用 SCS1:SCS0 = 00 定义的振荡器源运行
0 = 器件使用 SCS1:SCS0 = 01、 10 或 11 定义的振荡器源运行

bit 2 **未用：**读为 0

bit 1-0 **SCS1:SCS0：**系统时钟选择位
11 = 内部振荡器
10 = 主振荡器
01 =Timer1 振荡器
当 FOSC2 = 1 时：
00 = 主振荡器
当 FOSC2 = 0 时：
00 = 内部振荡器

注 1： 双速启动使能时复位值为 0，禁止时为 1。

2.7.1.1 系统时钟选择和 FOSC2 配置位

在所有形式的复位中，SCS 位都会被清零。在器件的默认配置中，这意味着 FOSC1:FOSC0 （也就是 HS 或 EC 模式的一种）定义的主振荡器用作器件复位时的主时钟源。

复位时的默认时钟配置可以随着 FOSC2 配置位的改变而改变。当 SCS1:SCS0 = 00 时此位会影响时钟源的选择。当 FOSC2 = 1（默认）时，不管何时 SCS1:SCS0 = 00，都选择由FOSC1:FOSC0定义的振荡器源。当FOSC2 = 0 时，不管何时 SCS1:SCS0 = 00，都选择 INTRC 振荡器。因为在复位时 SCS 位被清零，FOSC2 的设置也会更改复位时的默认振荡器模式。

不管 FOSC2 的设置如何，INTRC 总是会在器件上电时被使能。它将作为时钟源直到器件从存储器中装入了它的配置值。此时 FOSC 配置位被读取并选择了振荡器的工作模式。

注意主时钟或内部振荡器在任何给定条件下都会有两种位设置选项，这取决于 FOSC2 的设置。

2.7.2 振荡器转换

PIC18F97J60 系列器件包含在时钟源切换时防止时钟产生 “毛刺”的电路。在时钟切换时，系统时钟会有短暂的停顿。该停顿的时间长度是旧时钟源的两个周期加上新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

第 3.1.2 节“进入功耗管理模式”详细讨论了时钟切换。

2.8 功耗管理模式对各种时钟源的影响

当选定了 PRI_IDLE 模式后，指定的主振荡器会继续运行而不中断。对于所有其他功耗管理模式，使用 OSC1 引脚的振荡器会被禁止。OSC1 引脚（以及由振荡器使用的 OSC2 引脚）将会停止振荡。

在辅助时钟模式下（SEC_RUN和SEC_IDLE），Timer1 振荡器作为器件时钟源工作。如果需要，Timer1 振荡器也可以运行在所有功耗管理模式下为 Timer1 或 Timer3 提供时钟。

在 RC_RUN 和 RC_IDLE 模式下，由内部振荡器提供器件时钟源。无论是哪种功耗管理模式，31 kHz 的 INTRC 输出均可被直接用来提供时钟并且可被使能来支持多种特殊的功能部件 （关于 WDT、故障保护时钟监视器和双速启动的更多信息，请参见**第 24.2 节 “看门狗定时器（WDT）”**到**第 24.5 节“故障保护时钟监视器”**）。

如果选择了休眠模式，所有的时钟源都会被停止。因为休眠模式切断了所有晶体管的开启电流，休眠模式能实现最小的器件电流消耗 （仅泄漏电流）。

在休眠期间使能任何片上功能都将增加休眠时的电流消耗。要支持 WDT 工作，需要使能 INTRC。Timer1 振荡器可以用来为实时时钟提供时钟源。不需要器件时钟源的其他功能部件也可以工作（即，MSSP 从器件、PSP 和 INTx 引脚等）。在**第 27.2 节 “直流特性：掉电和供电电流”**中列出了可能显著增加电流消耗的外设。

2.9 上电延时

由两个定时器控制上电延时，这样大多数应用都无需外接复位电路。上电延时可以确保在器件电源稳定（常规环境下）和主时钟稳定工作之前器件保持在复位状态。关于上电延时的更多信息，请参见**第 4.5 节 “上电延时定时器 （PWRT）”**。

第一个定时器是上电延时定时器 （PWRT），在上电时它提供了固定的延迟时间 （表 27-12 中的参数 33）；它总是使能的。

第二个定时器是振荡器起振定时器 （OST），用于在晶振稳定前使芯片保持在复位状态（HS 模式）。OST 在计数 1024 个振荡周期后允许振荡器为器件提供时钟。

POR之后有一个TCSD间隔的延时（表27-12中的参数38），在此延时期间控制器为执行指令做准备。

表 2-3： 休眠模式下 OSC1 和 OSC2 引脚的状态

振荡器模式	OSC1 引脚	OSC2 引脚
EC 和 ECPLL	悬空，由外部时钟驱动	处于逻辑低电平 （时钟 /4 输出）
HS 和 HSPLL	反馈反相器被禁止，处于静止电平	反馈反相器被禁止，处于静止电平

注： 关于由休眠和 $\overline{\text{MCLR}}$ 复位引起的延时，请参见**第 4.0 节 “复位”**中的表 4-2。

3.0 功耗管理模式

PIC18F97J60 系列器件提供了只需通过管理 CPU 和外设的时钟源就可以管理功耗的功能。一般而言，较低的时钟频率和由时钟源驱动的电路数目的减少会使功耗降低。为了在应用中管理功耗，提供了三种主要的工作模式：

- 运行模式
- 空闲模式
- 休眠模式

这些模式定义了需要为器件的哪些部分提供时钟以及时钟的速度。运行和空闲模式可以使用三种时钟源（主时钟源、辅助时钟源或内部振荡器）中的任意一种；而休眠模式则不使用时钟源。

功耗管理模式包括几个由早期的 PIC® 器件提供的节省功耗的功能。其中之一就是其他 PIC18 器件也提供的时钟切换功能，该功能允许使用 Timer1 振荡器代替主振荡器。节省功耗的功能还包括所有 PIC 器件都提供的休眠模式，在该模式下，器件所有的时钟都停止。

3.1 选择功耗管理模式

选择功耗管理模式之前需要先做出两个决定：是否为 CPU 提供时钟源以及选择何种时钟源。IDLEN 位（OSCCON<7>）控制是否为 CPU 提供时钟源，而 SCS1:SCS0 位（OSCCON<1:0>）选择时钟源。表 3-1 总结了各个模式下的位设置、时钟源和受影响的模块。

3.1.1 时钟源

SCS1:SCS0 位允许为功耗管理模式在三个时钟源中任选其一。它们是：

- 主时钟，由 FOSC2:FOSC0 配置位定义
- 辅助时钟 （Timer1 振荡器）
- 内部振荡器

3.1.2 进入功耗管理模式

可以通过装载 OSCCON 寄存器从一种功耗管理模式切换到另一种功耗管理模式。 SCS1:SCS0 位选择时钟源并确定使用运行模式还是空闲模式。更改这些位会导致立即切换到一个新的时钟源（假定新时钟源正在运行）。此切换可能会引起时钟转换延时。第 **3.1.3 节 "时钟转换和状态指示"** 及其后续章节将会讨论这些问题。

执行SLEEP指令可以触发进入功耗管理空闲模式或休眠模式。最后实际进入哪个模式由 IDLEN 状态位决定。

更改功耗管理模式并不总是要求设置所有的位，而是取决于当前的模式和将要切换到的模式。通过在发出 SLEEP 指令之前更改振荡器选择位或更改 IDLEN 位可完成多种模式转换。如果已经正确配置了 IDLEN 位，可能只需执行 SLEEP 指令就可实现模式切换。

表 **3-1**： 功耗管理模式

模式	OSCCON 位		模块时钟		可用时钟和振荡器源
	IDLEN<7>[(1)]	SCS1:SCS0<1:0>	CPU	外设	
休眠	0	N/A	关闭	关闭	无——所有时钟被禁止
PRI_RUN	N/A	10	提供时钟	提供时钟	主时钟—— HS、EC、HSPLL 和 ECPLL ；这是正常的全功耗执行模式。
SEC_RUN	N/A	01	提供时钟	提供时钟	辅助时钟——Timer1 振荡器
RC_RUN	N/A	11	提供时钟	提供时钟	内部振荡器
PRI_IDLE	1	10	关闭	提供时钟	主时钟——HS、 EC、 HSPLL 和 ECPLL
SEC_IDLE	1	01	关闭	提供时钟	辅助时钟——Timer1 振荡器
RC_IDLE	1	11	关闭	提供时钟	内部振荡器

注 **1**： IDLEN 反映它在执行 SLEEP 指令时的值。

3.1.3 时钟转换和状态指示

在两个时钟源之间进行转换所需的时间长度是旧时钟源的两个周期与新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

以下两位用于指明当前的时钟源及其状态：OSTS（OSCCON<3>）和 T1RUN（T1CON<6>）。一般来说，在一个给定的功耗管理模式中，这两个位中只有一个位会置 1。当 OSTS 位置 1 时，表明由主时钟提供器件时钟。当 T1RUN 位置 1 时，表明由 Timer1 振荡器提供时钟源。如果这些位均不置 1，则由 INTRC 为器件提供时钟信号。

注：	执行 SLEEP 指令并不一定会将器件置于休眠模式。它只是作为触发条件，让器件进入休眠模式或一种空闲模式，具体何种模式由 IDLEN 位的设置决定。

3.1.4 多条 SLEEP 命令

使用 SLEEP 指令调用功耗管理模式时，具体进入何种模式在该指令执行那一刻由 IDLEN 位的设置决定。如果执行了另一条 SLEEP 指令，器件将进入由此时 IDLEN 位指定的功耗管理模式。如果 IDLEN 位已更改，器件将进入由新的设置指定的新的功耗管理模式。

3.2 运行模式

在运行模式中，内核和外设的时钟都是激活的。这些运行模式之间的区别就在于时钟源的不同。

3.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机的正常全功耗执行模式。除非使能了双速启动（详细信息，请参见**第 24.4 节 “双速启动”**），该模式也是器件复位后的默认模式。在此模式下，OSTS 位置 1（见**第 2.7.1 节 “振荡器控制寄存器”**）。

3.2.2 SEC_RUN 模式

SEC_RUN 模式与其他 PIC18 器件提供的“时钟切换”功能兼容。在此模式下，CPU 和外设将 Timer1 振荡器作为时钟源。这允许用户在使用高精度时钟源的情况下仍可获得较低的功耗。

通过将 SCS1:SCS0 位置为 `01` 进入 SEC_RUN 模式。器件时钟源被切换到 Timer1 振荡器（见图 3-1），主振荡器被关闭，T1RUN 位（T1CON<6>）被置 1 并且 OSTS 位被清零。

注：	Timer1 振荡器应该在进入 SEC_RUN 模式之前就已经运行了。如果在 SCS1:SCS0 位被置为 `01` 时 T1OSCEN 位没有置 1，就不会进入 SEC_RUN 模式。如果 Timer1 振荡器已经被使能，但没有开始运行，器件时钟将会延时直到该振荡器起振。在这种情况下，最初的振荡器运行很不稳定，可能会导致无法预料的结果。

在从 SEC_RUN 模式转换到 PRI_RUN 模式期间，外设和 CPU 继续使用 Timer1 振荡器作为时钟源，直到主时钟启动。当主时钟准备好以后，时钟切换回主时钟（见图 3-2）。当时钟切换完成后，T1RUN 位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。这种唤醒不会影响 IDLEN 和 SCS 位。Timer1 振荡器继续运行。

图 **3-1**： 进入 **SEC_RUN** 模式的转换时序

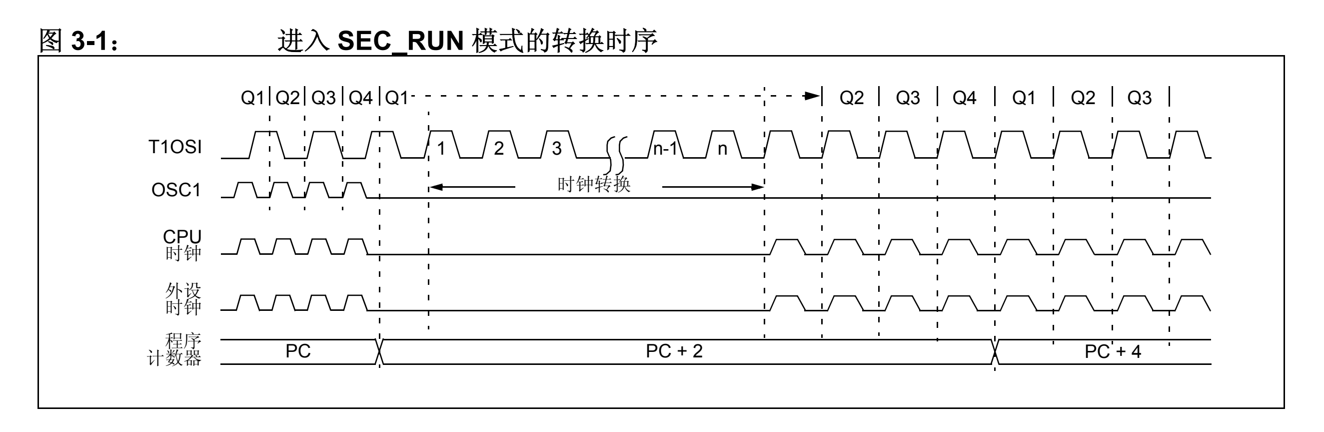
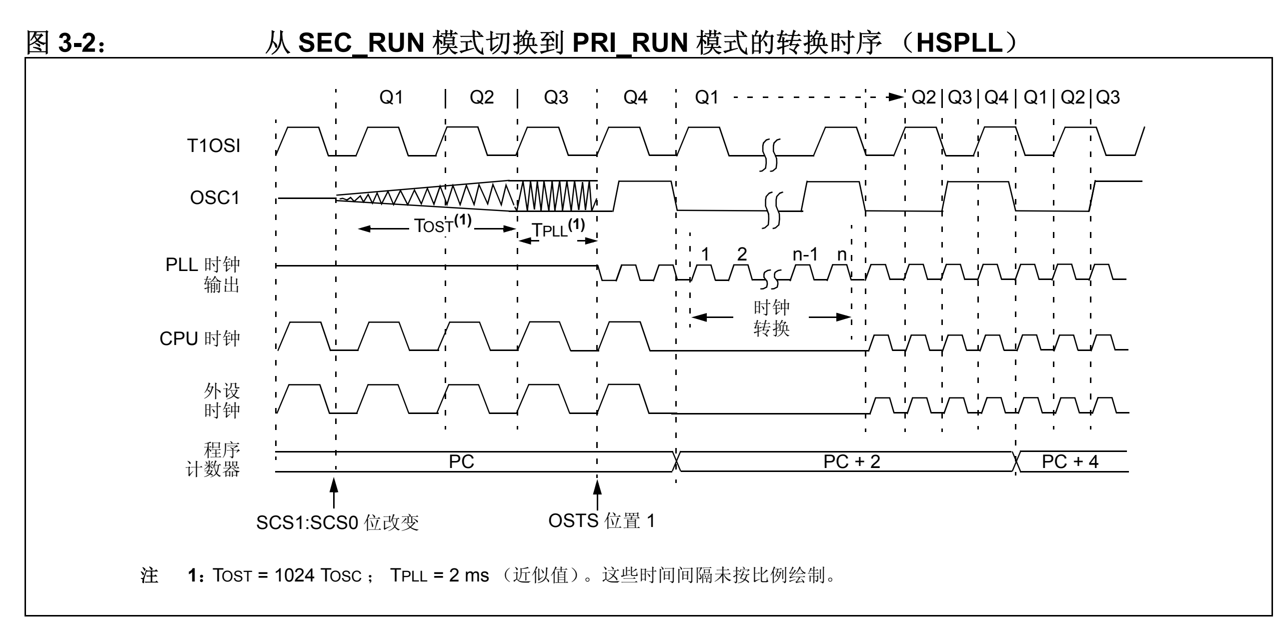


图 **3-2**： 从 **SEC_RUN** 模式切换到 **PRI_RUN** 模式的转换时序 （**HSPLL**）



注 1：T_{OST} = 1024 T_{OSC}； T_{PLL} = 2 ms （近似值）。这些时间间隔未按比例绘制。

3.2.3 RC_RUN 模式

在 RC_RUN 模式下，内部振荡器作为 CPU 和外设的时钟源；主时钟关闭。此模式是在代码执行期间所有运行模式中最节省功耗的运行模式。它非常适用于对定时精度要求不高或者不是一直需要高速时钟的应用。

通过将 SCS 置为 11 可以进入此模式。当将时钟源切换到INTRC时（见图 3-3），主振荡器将被关闭并且OSTS 位被清零。

在从 RC_RUN 模式转换到 PRI_RUN 模式期间，在主时钟处于启动状态时，器件将继续使用 INTRC 作为时钟源。当主时钟准备好以后，时钟切换回主时钟（见图 3-4）。当时钟切换完成后，OSTS 位被置 1 并且由主时钟提供器件时钟。这种切换不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器，INTRC 源将继续运行。

图 3-3： 到 RC_RUN 模式的转换时序

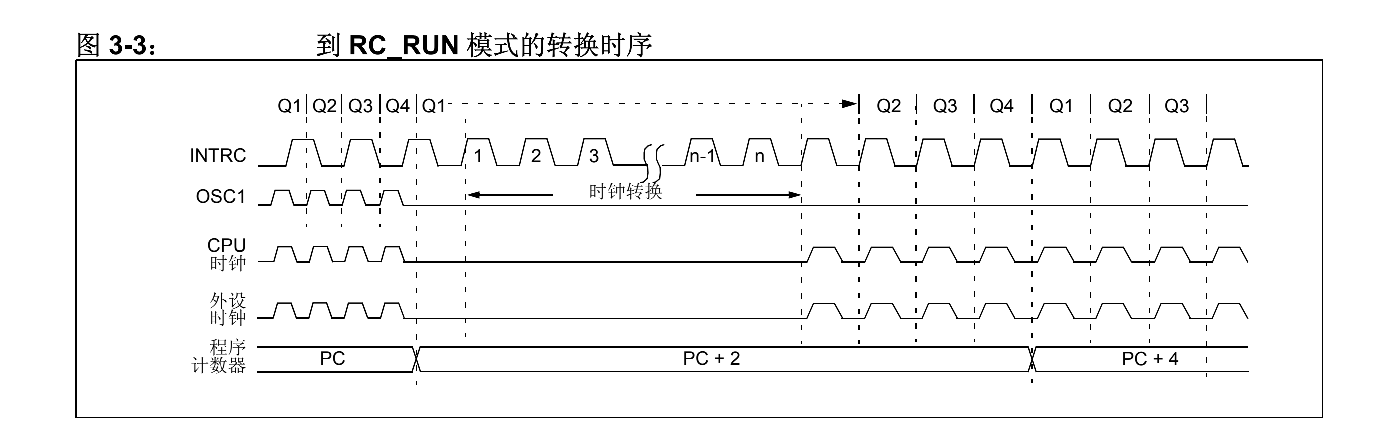
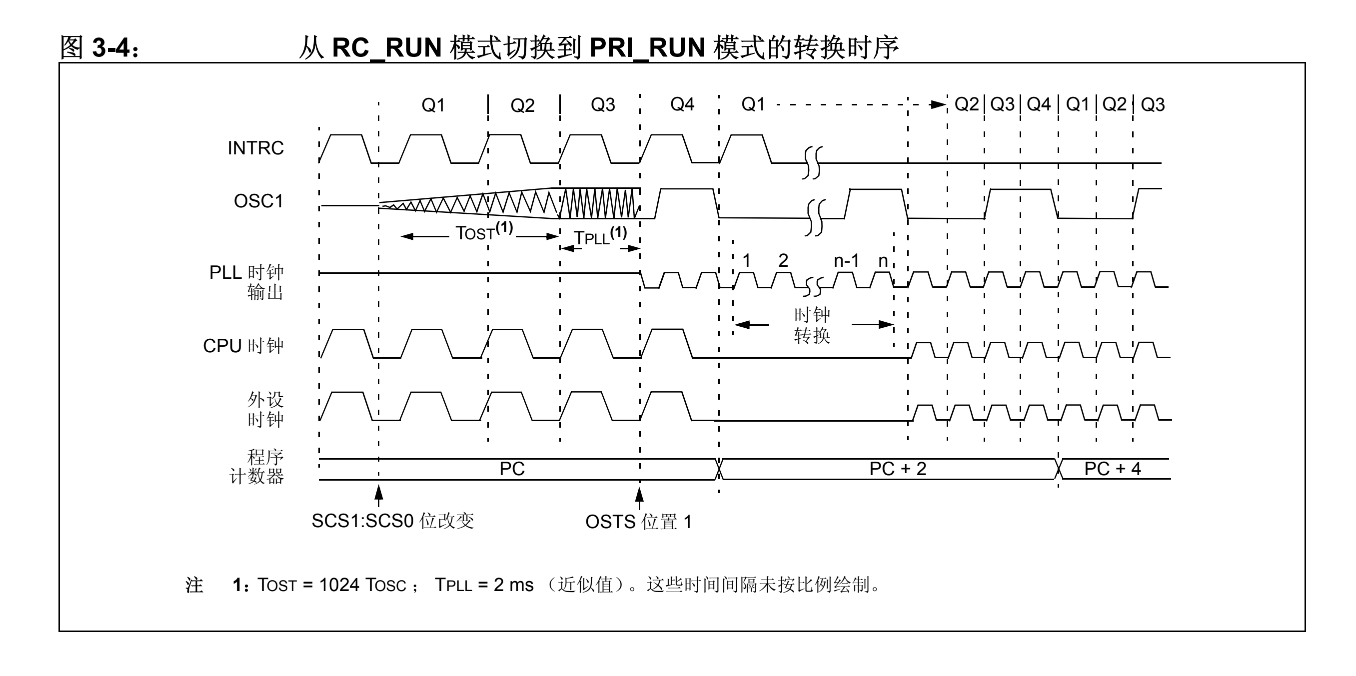


图 3-4： 从 RC_RUN 模式切换到 PRI_RUN 模式的转换时序



 超前信息

3.3 休眠模式

功耗管理休眠模式和所有其他PIC器件提供的传统休眠模式相同。通过清零 IDLEN 位（器件复位时的默认状态）并执行 SLEEP 指令即可进入此模式。这将关闭所选择的振荡器（见图 3-5），并将所有的时钟源状态位清零。

从其他模式进入休眠模式不需要切换时钟。这是因为单片机一旦进入休眠模式就不需要时钟了。如果选择了 WDT，INTRC 时钟源将继续工作。如果使能了 Timer1 振荡器，它也将继续运行。

当在休眠模式中发生唤醒事件时（通过中断、复位或 WDT 超时），在时钟源（通过 SCS1:SCS0 位选择）准备好之前器件将没有时钟源（见图 3-6），或者如果使能了双速启动或故障保护时钟监视器，它将使用内部振荡器作为时钟源（见第 **24.0** 节 **"CPU 的特殊功能"**）。在这两种情况下，当由主时钟提供器件时钟时，OSTS 位将置 1。这种唤醒不会影响 IDLEN 和 SCS 位。

3.4 空闲模式

空闲模式允许在外设继续工作的同时有选择地关闭单片机的CPU。选择特定的空闲模式允许用户进一步管理功耗。

如果在执行 SLEEP 指令时，IDLEN 位置 1，外设将使用由 SCS1:SCS0 位选择的时钟作为时钟源，而 CPU 没有时钟源。时钟源状态位不受影响。将 IDLEN 置 1 并执行 SLEEP 指令可以从给定的运行模式快速切换到相应的空闲模式。

如果选择了 WDT，INTRC 时钟源将继续工作。如果使能了 Timer1 振荡器，它也将继续运行。

由于 CPU 没有执行指令，器件只能通过中断、WDT 超时或复位从空闲模式退出。当发生唤醒事件时，CPU 会在其准备好执行代码前延时一个 T_{CSD} 间隔（表 27-12 中的参数 38）。当 CPU 开始执行代码时，它将沿用当前空闲模式所使用的时钟源。例如，当从 RC_IDLE 模式唤醒时，将使用内部振荡器模块为 CPU 和外设提供时钟（即 RC_RUN 模式）。这种唤醒不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式中时，WDT 超时会导致 WDT 唤醒并进入当前由 SCS1:SCS0 位指定的运行模式。

图 3-5： 进入休眠模式的转换时序

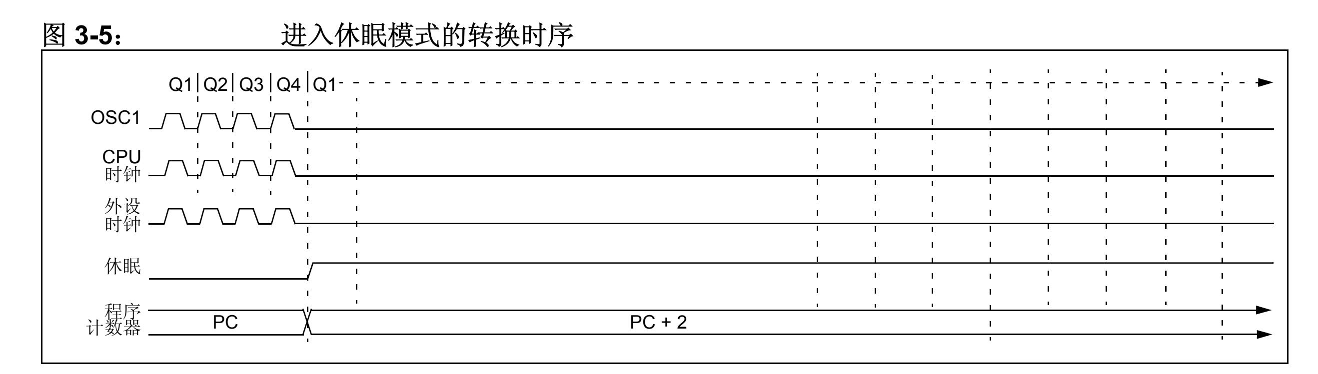
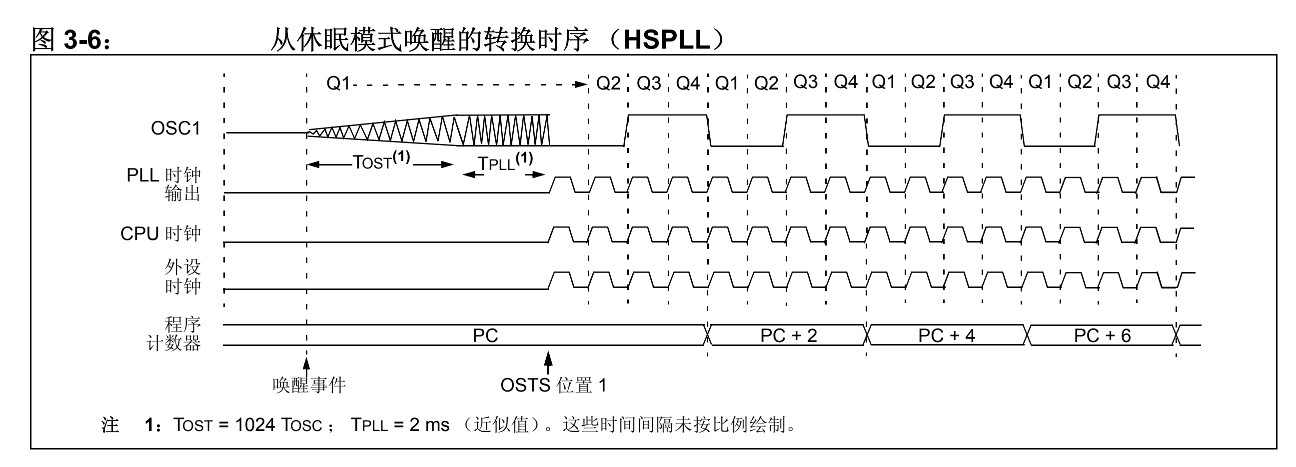


图 3-6： 从休眠模式唤醒的转换时序（**HSPLL**）



注 1：T_{OST} = 1024 T_{OSC}；T_{PLL} = 2 ms（近似值）。这些时间间隔未按比例绘制。

3.4.1 PRI_IDLE 模式

在三种低功耗空闲模式中，只有该模式不会禁止主器件时钟。由于时钟源不需要“热身”或是从其他振荡器转换过来，选用此模式可以使对时间要求较高的应用以最快的速度恢复器件运行，并使用较精确的主时钟源。

可以通过将 IDLEN 位置 1 并执行 SLEEP 指令以实现从 PRI_RUN 模式进入 PRI_IDLE 模式。如果器件处于另一种运行模式，首先将 IDLEN 位置 1，然后将 SCS 位置为 10 并执行 SLEEP。虽然 CPU 已被禁止，但外设仍继续使用由FOSC1:FOSC0配置位指定的主时钟源为其提供时钟信号。 OSTS 位保持置 1 （见图 3-7）。

当发生唤醒事件时，由主时钟源为 CPU 提供时钟。在唤醒事件和代码执行开始之间需要一个 TCSD 间隔的延时。该延时用来让 CPU 做好执行指令的准备。在唤醒之后，OSTS 位保持置 1 状态。这种唤醒不会影响 IDLEN 和 SCS 位 （见图 3-8）。

3.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式下， CPU 被禁止，但外设继续将 Timer1 振荡器作为时钟源。可以通过将 IDLEN 位置 1 并执行 SLEEP 指令从 SEC_RUN 模式进入此模式。如果器件处于另一种运行模式，首先将 IDLEN 位置 1，然后将 SCS1:SCS0 置为 01 并执行 SLEEP。当时钟源切换到 Timer1 振荡器时，主振荡器关闭， OSTS 位被清零并且 T1RUN 位置 1。

当唤醒事件发生时，外设继续将 Timer1 振荡器作为时钟源。唤醒事件发生后经过一个 TCSD 时间间隔， CPU 开始执行代码并使用 Timer1 振荡器作为其时钟源。这种唤醒不会影响 IDLEN 和 SCS 位。Timer1 振荡器继续运行 （见图 3-8）。

注： Timer1 振荡器应该在进入 SEC_IDLE 模式之前就已经运行了。如果执行 SLEEP 指令时 T1OSCEN 位没有置 1，就会忽略 SLEEP 指令并不会进入 SEC_IDLE 模式。如果使能了 Timer1 振荡器，但它尚未运行，外设时钟将会延时直到该振荡器起振。在这种情况下，最初的振荡器运行很不稳定，可能会导致无法预料的结果。

图 3-7： 进入空闲模式的转换时序

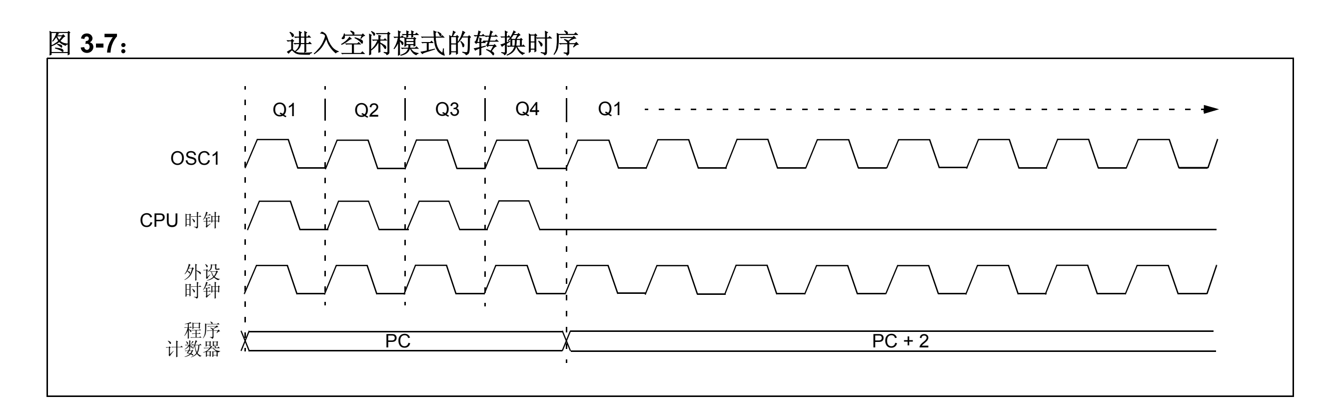
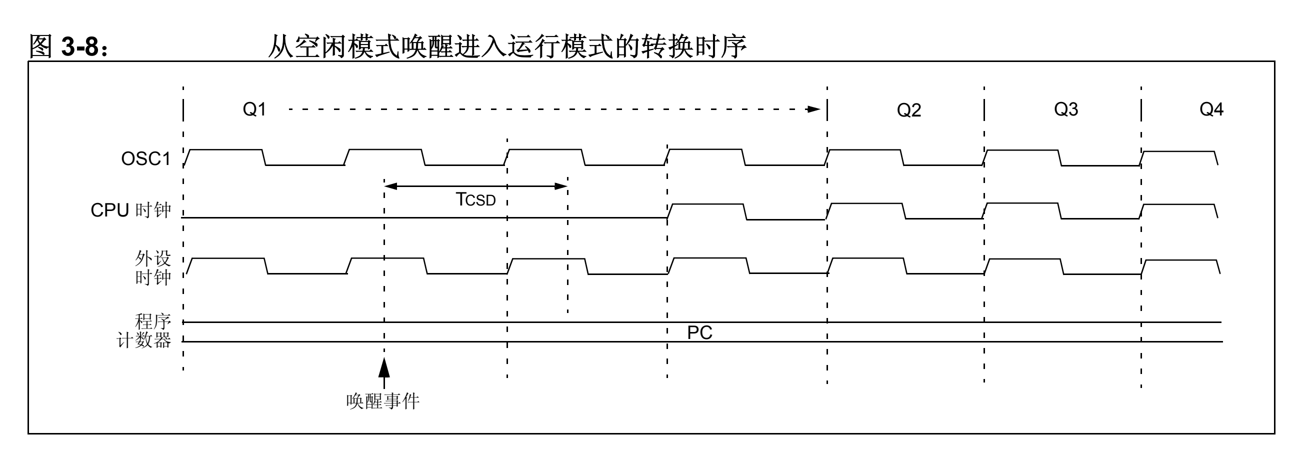


图 3-8： 从空闲模式唤醒进入运行模式的转换时序



3.4.3 RC_IDLE 模式

RC_IDLE 模式禁止 CPU，但仍继续由内部振荡器为外设提供时钟。该模式允许在空闲期间对功耗进行控制。

可以通过将IDLEN位置1并执行SLEEP指令从RC_RUN模式进入此模式。如果器件处于另一种运行模式，首先将 IDLEN 位置 1，然后清零 SCS 位并执行 SLEEP。当时钟源切换到 INTRC 时，主振荡器被关闭， OSTS 位被清零。

当唤醒事件发生时，外设继续将 INTRC 作为时钟源。在唤醒事件后的 TCSD 间隔之后，CPU 开始执行代码并使用 INTRC 作为时钟源。这种唤醒不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器，INTRC 源将继续运行。

3.5 退出空闲和休眠模式

由中断、复位或 WDT 超时触发从休眠模式或任何空闲模式的退出。本节将讨论从功耗管理模式退出的触发方式。在每种功耗管理模式章节中我们已经讨论过其时钟源子系统的作用 （见**第 3.2 节 “运行模式”**、**第 3.3 节 “休眠模式”**和**第 3.4 节 “空闲模式”**）。

3.5.1 通过中断退出

任何可用的中断源都可导致器件从空闲模式或休眠模式退出到运行模式。要使能此功能，必须通过将对应 INTCON 或 PIE 寄存器中的中断源允许位置 1 来允许中断源。当相应的中断标志位置 1 时，触发退出操作。

当使用中断从空闲或休眠模式退出时，如果 GIE/GIEH 位 （INTCON<7>）置 1，程序就会跳转到中断向量处执行代码。否则代码就会顺序执行 （见**第 9.0 节 “中断”**）。

唤醒事件之后需要一个固定的 TCSD 间隔的延时，器件才会退出休眠和空闲模式。CPU 需要此延时来准备执行代码。在延时后的第一个时钟周期重新开始执行指令。

3.5.2 通过 WDT 超时退出

根据 WDT 超时发生时器件所处的不同功耗管理模式会引发不同的操作。

如果器件不在执行代码 （所有空闲模式和休眠模式），超时将导致从功耗管理模式退出（见**第 3.2 节“运行模式”**和**第 3.3 节 “休眠模式”**）。如果器件正在执行代码 （所有运行模式），超时将导致 WDT 复位 （见**第 24.2 节 “看门狗定时器 （WDT）”**）。

WDT 定时器和后分频器可由以下任一事件清零：

- 执行 SLEEP 或 CLRWDT 指令
- 当前选择的时钟源失效 （如果使能了故障保护时钟监视器）

3.5.3 通过复位退出

通过复位退出空闲或休眠模式会自动强制器件使用 INTRC 运行。

3.5.4 在没有振荡器起振延时的情况下退出

从某些功耗管理模式退出完全不需要 OST 延时。有以下两种情形：

- 主时钟源不停止的 PRI_IDLE 模式
- 主时钟源为 EC 或 ECPLL 模式

在这些情况下，主时钟源不需要振荡器起振延时，因为它已经在运行（PRI_IDLE），或者它本来就不需要振荡器起振延时（EC）。但是，当器件退出休眠和空闲模式时，在唤醒事件之后仍然需要一个固定的 TCSD 间隔的延时，以便让 CPU 准备好执行代码。在延时后的第一个时钟周期重新开始执行指令。

注：

4.0 复位

PIC18F97J60 系列器件有以下几种不同的复位方式：

a） 上电复位 （POR）
b） 正常工作状态下的 $\overline{\text{MCLR}}$ 复位
c） 功耗管理模式下的 $\overline{\text{MCLR}}$ 复位
d） 看门狗定时器 （WDT）复位 （执行程序期间）
e） 欠压复位 （BOR）
f） RESET 指令复位
g） 堆栈满复位
h） 堆栈下溢复位

本节将讨论由 $\overline{\text{MCLR}}$、 POR 和 BOR 产生的复位，并涉及各种起振定时器的工作方式。堆栈复位事件将在**第 5.1.6.4 节 “堆栈满和下溢复位”**中讨论。 WDT 复位将在**第 24.2 节 “看门狗定时器 （WDT）”**中讨论。

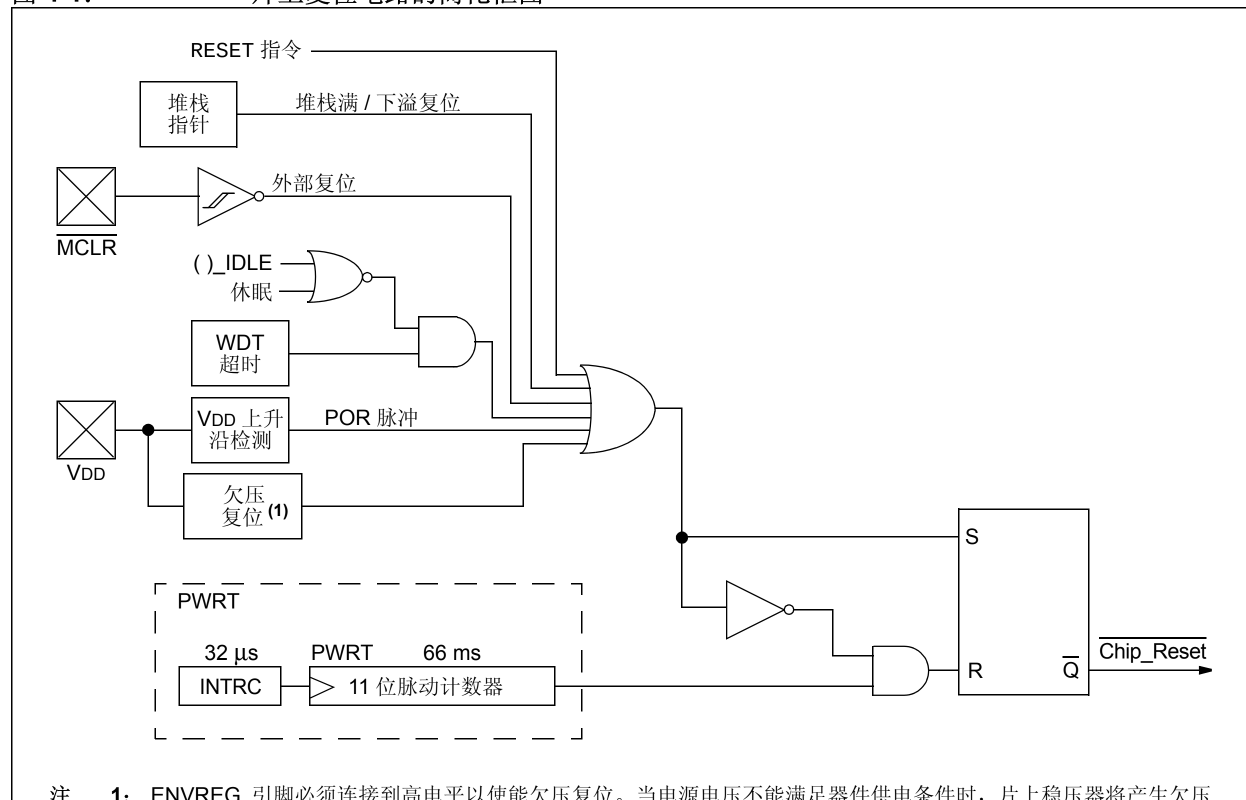
图 4-1 给出了片上复位电路的简化框图。

4.1 RCON 寄存器

通过 RCON 寄存器（寄存器 4-1）跟踪器件复位事件。该寄存器的低 5 位表明是否已经发生了特定的复位事件。在大多数情况下，只能通过事件将这些位置 1，而且必须在事件发生后由应用程序将它们清零。需要读取所有这些标志位来确定刚发生的复位的类型。在**第 4.6 节 “寄存器的复位状态”**中对此进行了更详细的说明。

RCON 寄存器还有设置中断优先级的控制位（IPEN）。在**第 9.0 节 “中断”**中讨论了中断优先级。

图 4-1： 片上复位电路的简化框图



注 1： ENVREG 引脚必须连接到高电平以使能欠压复位。当电源电压不能满足器件供电条件时，片上稳压器将产生欠压复位。

寄存器 4-1： RCON：复位控制寄存器

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **IPEN：**中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit 6-5 未用：读为 0

bit 4 **$\overline{\text{RI}}$：**RESET 指令标志位
1 = 未执行 RESET 指令 （只能由固件置 1）
0 = 已执行了 RESET 指令，导致器件复位 （必须在发生复位后由软件置 1）

bit 3 **$\overline{\text{TO}}$：**看门狗超时标志位
1 = 通过上电、 CLRWDT 指令或 SLEEP 指令置 1
0 = 发生了 WDT 超时溢出

bit 2 **$\overline{\text{PD}}$：**掉电检测标志位
1 = 通过上电或 CLRWDT 指令置 1
0 = 通过执行 SLEEP 指令置 1

bit 1 **$\overline{\text{POR}}$：**上电复位状态位
1 = 未发生上电复位 （只能由固件置 1）
0 = 已发生上电复位 （必须在发生上电复位后由软件置 1）

bit 0 **$\overline{\text{BOR}}$：**欠压复位状态位
1 = 未发生欠压复位 （只能由固件置 1）
0 = 已发生欠压复位 （必须在发生欠压复位后由软件置 1）

注 **1：**建议在检测到上电复位后，将 $\overline{\text{POR}}$ 位置 1，以便继续检测后续的上电复位。
2：如果禁止了片上稳压器， $\overline{\text{BOR}}$ 则总是保持为 0。更多信息，请参见**第 4.4.1 节 “检测 BOR”**。
3：当 $\overline{\text{BOR}}$ 为 0 并且 $\overline{\text{POR}}$ 为 1 时（假定在上电复位之后立即用软件将 $\overline{\text{POR}}$ 置 1），可以说已发生了欠压复位。

4.2 主复位（MCLR）

MCLR 引脚提供触发硬件外部复位器件的方法。将该引脚拉低可以产生复位信号。PIC18 扩展的单片机器件在 MCLR 复位路径上有一个噪声滤波器，该滤波器可以检测并滤除小的干扰脉冲。

任何内部复位，包括 WDT 复位，均不能将 MCLR 引脚驱动为低电平。

4.3 上电复位（POR）

只要当 VDD 上升超过某个门限时，就会在片上产生上电复位条件。这使得 VDD 达到满足器件正常工作的数值时，器件会以初始化状态启动。

为了利用 POR 电路，需要将 MCLR 引脚通过一个电阻（阻值范围为 1 kΩ 到 10 kΩ）连接到 VDD。这样可以省去产生上电复位延时通常所需的外部 RC 元件。VDD 的最小上升速率已指定（参数 D004）。上升速率缓慢的情况，请参见图 4-2。

当器件开始正常工作（即，退出复位状态）时，器件的工作参数（电压、频率和温度等）必须得到满足，以确保其正常工作。如果不满足这些条件，那么器件必须保持在复位状态，直到满足工作条件为止。

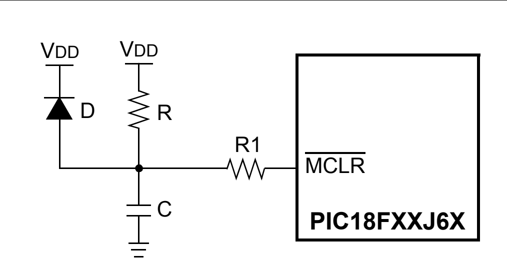
POR 事件由 POR 位（RCON<1>）捕获。每当发生上电复位时，该位的状态就会被置为 0；任何其他复位事件均不能改变它。任何硬件事件均不能将 POR 复位为 1。要捕获多个事件，用户必须在上电复位之后用软件手动将该位复位为 1。

4.4 欠压复位（BOR）

当内部稳压器被使能时（ENVREG 引脚连接到 VDD），PIC18F97J60 系列器件就具备了简单的 BOR 功能。只要 VDD 低于 VBOR（参数 D005）的时间大于 TBOR（参数 35）就会复位器件。如果 VDD 降到 VBOR 以下的时间小于 TBOR，器件是否发生复位不确定。芯片将保持欠压复位状态，直至 VDD 电压上升到 VBOR 以上。

一旦发生BOR，上电延时定时器将芯片保持在复位状态的时间就是 TPWRT（参数 33）。如果在上电延时定时器运行过程中，VDD 电压降到 VBOR 以下，芯片将重新回到欠压复位状态并且初始化上电延时定时器。一旦 VDD 电压上升到 VBOR 以上，上电延时定时器将重新执行延时。

图 4-2： 外部上电复位电路（VDD 缓慢上电的情况）



注 1：仅当 VDD 上电速率过慢时才需要外部上电复位电路。二极管 D 有助于在 VDD 掉电时使电容迅速放电。

2：建议 R < 40 kΩ，确保电阻 R 两端压降符合器件的电气规范。

3：R1 ≥ 1 kΩ 将限制任何电流从外部电容 C 流入 MCLR，以避免由于静电放电（Electrostatic Discharge，ESD）或电过载（Electrical Overstress，EOS）导致 MCLR/VPP 引脚损坏。

4.4.1 检测 BOR

BOR 位在欠压复位或上电复位事件时总是复位为 0。因此只通过读 BOR 位的状态很难确定是否发生过欠压复位事件。更可靠的方法是同时检查 POR 和 BOR 的状态。假定在发生任何上电复位事件后，POR 位被立即用软件复位为 1。如果 BOR 为 0 同时 POR 为 1，那么就可以断定已经发生了欠压复位事件。

如果禁止稳压器，也会禁止欠压复位功能。在这种情况下，不能使用 BOR 位来确定欠压复位事件。上电复位事件仍会将 BOR 位清零。

4.5 上电延时定时器（PWRT）

PIC18F97J60 系列器件具有片上上电延时定时器（PWRT）以帮助稳定上电复位过程。PWRT 总是使能的。其主要功能是确保在代码执行之前，器件的电压是稳定的。

PIC18F97J60 系列器件的上电延时定时器（PWRT）是一个 11 位计数器，它使用 INTRC 时钟源作为时钟输入。该定时器可产生大约 2048 x 32 µs = 66 ms 的时间间隔。PWRT 计数期间，器件保持在复位状态。

上电延时时间取决于 INTRC 时钟，并且由于温度和工艺的不同，不同器件的延迟时间也将各不相同。详情请参见直流参数 33。

4.5.1 延时时序

在 POR 脉冲被清零后，启动 PWRT 延时。总延迟时间将取决于 PWRT 的状态。图 4-3、图 4-4、图 4-5 和图 4-6 都说明了在使能上电延时定时器时的延时序列。

由于延时是由 POR 脉冲触发的，因此如果 $\overline{MCLR}$ 保持足够长时间的低电平，PWRT 将结束。将 MCLR 电平拉高后器件将立即开始执行代码（图 4-5）。这对于测试或同步多个并行工作的 PIC18FXXJ6X 器件是非常有用的。

图 4-3： 上电延时时序（$\overline{MCLR}$ 连接到 VDD，VDD 电压上升时间 < TPWRT）

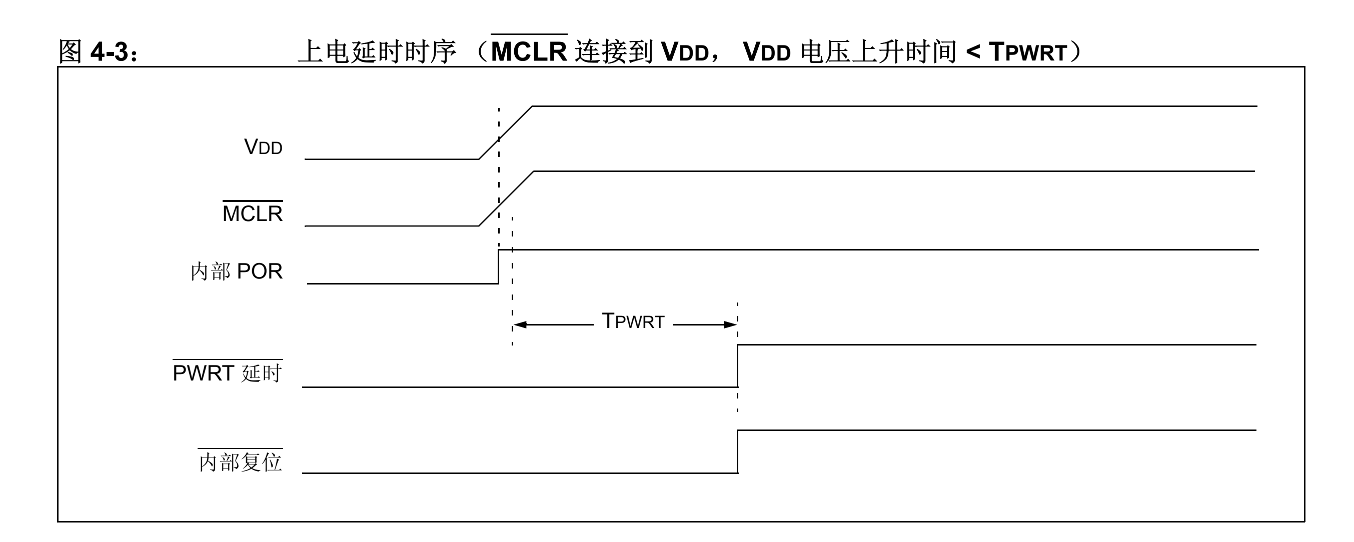


图 4-4： 上电延时时序（$\overline{MCLR}$ 未连接到 VDD）：情形 1

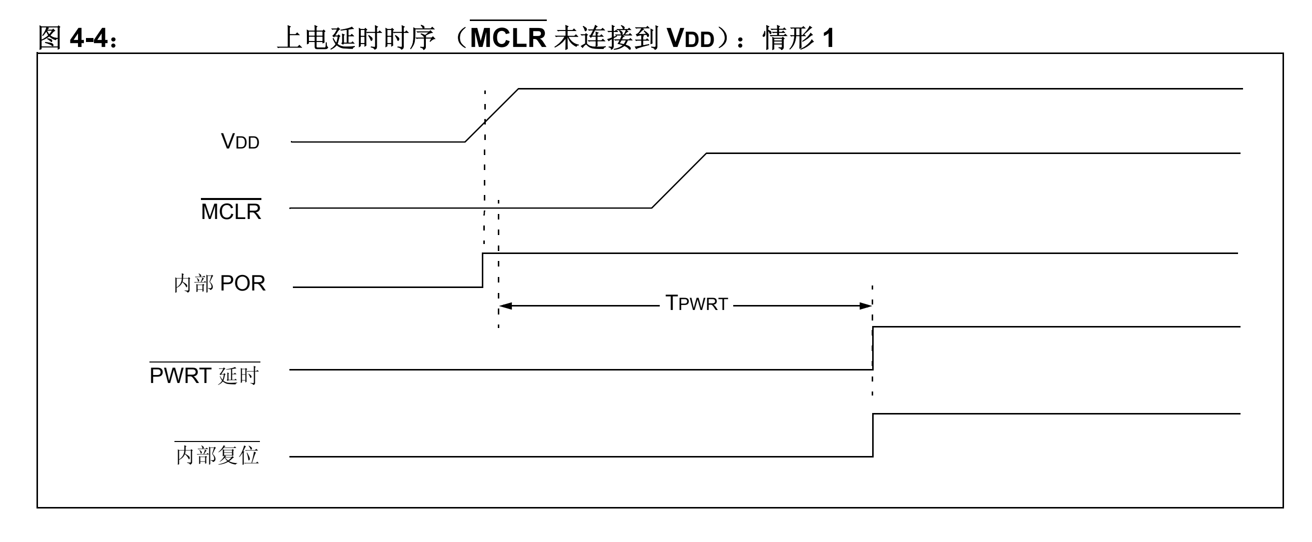


图 **4-5：** 上电延时时序（**MCLR** 未连接到 **VDD**）：情形 **2**

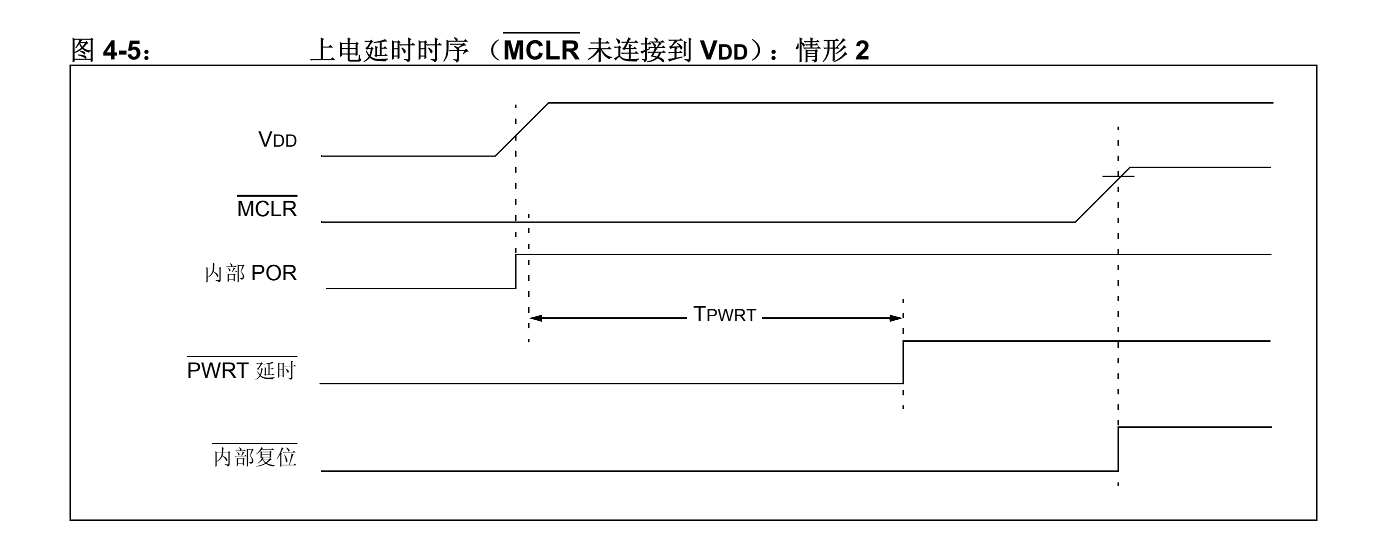
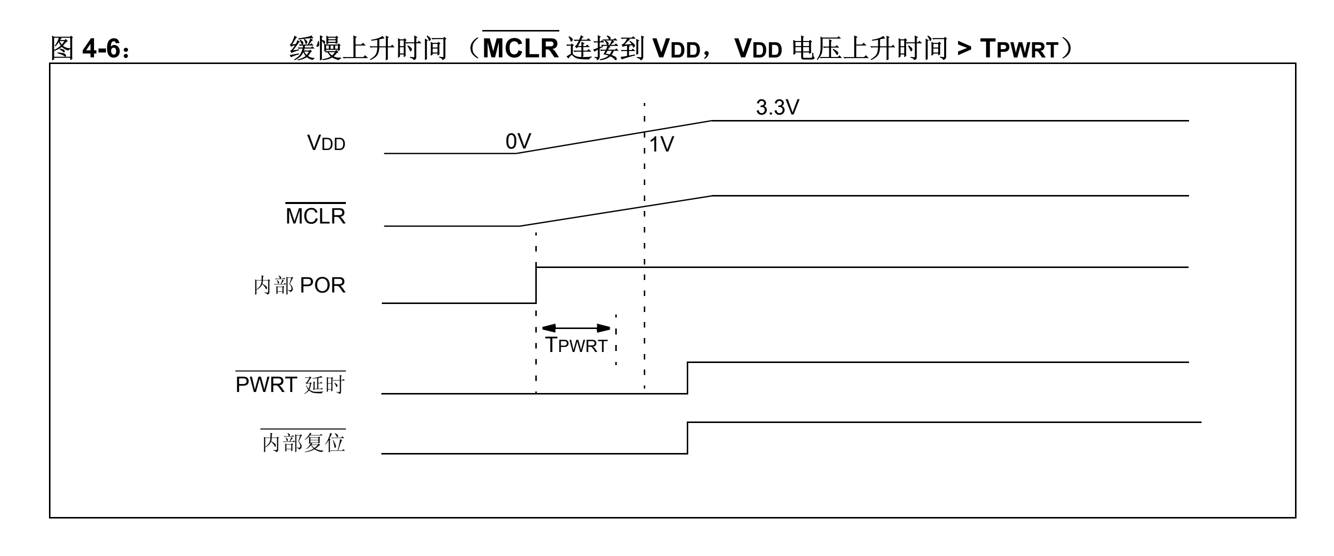


图 **4-6：** 缓慢上升时间（**MCLR** 连接到 **VDD**， **VDD** 电压上升时间 **> TPWRT**）



4.6 寄存器的复位状态

大多数寄存器不受复位的影响。在 POR 时这些寄存器的状态不确定，而在其他复位时它们的状态不变。而剩余寄存器则根据不同的复位类型被强制为 “复位状态”。

大多数寄存器不受 WDT 唤醒的影响，这是因为 WDT 唤醒被视为对正常工作的恢复。如表 4-1 所示，RCON 寄存器中的状态位：$\overline{RI}$、$\overline{TO}$、$\overline{PD}$、$\overline{POR}$ 和 $\overline{BOR}$，在不同的复位情形中会分别被置 1 或清零。可在软件中使用这些状态位判断复位的性质。

表 4-2 描述了所有特殊功能寄存器的复位状态。可以将这些复位状态分类为上电和欠压复位、主复位、WDT 复位以及 WDT 唤醒。

表 4-1： RCON 寄存器的状态位、含义以及初始化状态

条件	程序计数器 (1)	RCON 寄存器					STKPTR 寄存器	
		$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
上电复位	0000h	1	1	1	0	0	0	0
RESET 指令	0000h	0	u	u	u	u	u	u
欠压复位	0000h	1	1	1	u	0	u	u
功耗管理运行模式下的 $\overline{MCLR}$ 复位	0000h	u	1	u	u	u	u	u
功耗管理空闲和休眠模式下的 $\overline{MCLR}$ 复位	0000h	u	1	0	u	u	u	u
全功耗或功耗管理运行模式期间的 WDT 超时	0000h	u	0	u	u	u	u	u
全功耗执行期间的 $\overline{MCLR}$ 复位	0000h	u	u	u	u	u	u	u
堆栈满复位 （STVREN = 1）	0000h	u	u	u	u	u	1	u
堆栈下溢复位 （STVREN = 1）	0000h	u	u	u	u	u	u	1
堆栈下溢错误 （不是真正的复位，STVREN = 0）	0000h	u	u	u	u	u	u	1
功耗管理空闲或休眠模式下的 WDT 超时	PC + 2	u	0	0	u	u	u	u
通过中断从功耗管理模式退出	PC + 2	u	u	0	u	u	u	u

图注： u = 不变

注 1： 当器件被中断唤醒且 GIEH 或 GIEL 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

 超前信息

表 4-2：　所有寄存器的初始化状态

寄存器	适用器件			上电复位，欠压复位	$\overline{MCLR}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
TOSU	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---0 uuuu(1)
TOSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTINC0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTDEC0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PREINC0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PLUSW0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
FSR0H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTINC1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTDEC1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PREINC1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PLUSW1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
FSR1H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTINC2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTDEC2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PREINC2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PLUSW2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
FSR2H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注：　u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注　1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
STATUS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TMR0L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0--- q-00	0--- q-00	u--- q-uu
ECON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 00--	0000 00--	uuuu uu--
WDTCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- ---0	---- ---0	---- ---u
RCON[4]	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0--1 1100	0--q qquu	u--u qquu
TMR1H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	1111 1111
T2CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP1STAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ADRESH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-00 0000	0-00 0000	u-uu uuuu
ADCON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
ADCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-00 0000	0-00 0000	u-uu uuuu
CCPR1H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CCPR3H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP3CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP1AS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CMCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0111	0000 0111	uuuu uuuu
TMR3H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
PSPCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 ----	0000 ----	uuuu ----
SPBRG1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXSTA1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 000x	0000 000x	uuuu uuuu
EECON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- ----	---- ----	---- ----
EECON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 x00-	---0 x00-	---u uuu-
IPR3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
PIR3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(3)
PIE3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
IPR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1-11	1111 1-11	uuuu u-uu
PIR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0-00	0000 0-00	uuuu u-uu(3)
PIE2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0-00	0000 0-00	uuuu u-uu
IPR1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(3)
PIE1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MEMCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-00 --00	0-00 --00	u-uu --uu
OSCTUNE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 ----	0000 ----	uuuu ----
TRISJ	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 ----	--11 ----	--uu ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---1 ----	---1 ----	---u ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---1 1111	---1 1111	---u uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 111-	1111 111-	uuuu uuu-
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 1111	--11 1111	--uu uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -111	---- -111	---- -uuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 1111	--11 1111	--uu uuuu
LATJ	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx ----	--uu ----	--uu ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
LATG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x ----	---u ----	---u ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x xxxx	---u uuuu	---u uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxx-	uuuu uuu-	uuuu uuu-
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx xxxx	--uu uuuu	--uu uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -xxx	---- -uuu	---- -uuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	00xx xxxx	00uu uuuu	uuuu uuuu
PORTJ	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx ----	--uu ----	--uu ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x ----	---u ----	---u ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x xxxx	---u uuuu	---u uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	111x xxxx	111u uuuu	uuuu uuuu
PORTF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	x000 000-	x000 000-	uuuu uuu-
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	x000 000-	x000 000-	uuuu uuu-
PORTE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx xxxx	--uu uuuu	--uu uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -xxx	---- -uuu	---- -uuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-0x 0000	0-0u 0000	u-uu uuuu
SPBRGH1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0100 0-00	01-0 0-00	uu-u u-uu
SPBRGH2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
BAUDCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0100 0-00	01-0 0-00	uu-u u-uu
ERDPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 1010	---0 1010	---u uuuu
ERDPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 0101	1111 0101	uuuu uuuu
ECCP1DEL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TMR4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PR4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	1111 1111
T4CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
CCPR4H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR4L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
CCP4CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
CCPR5H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR5L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP5CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
SPBRG2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0010	0000 0010	uuuu uuuu
RCSTA2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 000x	0000 000x	uuuu uuuu
ECCP3AS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP3DEL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP2AS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2BUF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP2ADD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2STAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2CON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2CON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDATA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
EIR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0-00	-000 0-00	-uuu u-uu
ECON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	100- ----	100- ----	uuu- ----
ESTAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-0-0 -000	-0-0 -000	-u-u -uuu
EIE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0-00	-000 0-00	-uuu u-uu
EDMACSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMACSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMADSTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EDMADSTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMANDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EDMANDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMASTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EDMASTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ERXWRPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
ERXWRPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ERXRDPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0101	---0 0101	---u uuuu
ERXRDPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1010	1111 1010	uuuu uuuu
ERXNDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---1 1111	---1 1111	---u uuuu
ERXNDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
ERXSTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0101	---0 0101	---u uuuu
ERXSTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1010	1111 1010	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注 **1：** 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
ETXNDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
ETXNDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ETXSTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
ETXSTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EWRPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EWRPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPKTCNT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ERXFCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1010 0001	1010 0001	uuuu uuuu
EPMOH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EPMOL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMCSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMCSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM7	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM6	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM5	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT7	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT6	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT5	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIRDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIRDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIWRH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIWRL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIREGADR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
MICMD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- --00	---- --00	---- --uu
MICON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0--- ----	0--- ----	u--- ----
MAMXFLH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0110	0000 0110	uuuu uuuu
MAMXFLL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MACLCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 0111	--11 0111	--uu uuuu
MACLCON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- 1111	---- 1111	---- uuuu

图注： u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

表 **4-2**： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒器件
MAIPGH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
MAIPGL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
MABBIPG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
MACON4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 --00	-000 --00	-uuu --uu
MACON3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MACON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EPAUSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0001 0000	0001 0000	000u uuuu
EPAUSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EFLOCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -000	---- -000	---- -uuu
MISTAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- 0000	---- 0000	---- uuuu
MAADR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR6	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR5	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用位 （读为 0）， q = 值取决于具体条件。阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

4： 具体条件下的复位值，请参见表 4-1。

注：

5.0 存储器构成

PIC18 闪存单片机器件有两种类型的存储器：

- 程序存储器
- 数据 RAM

由于是哈佛架构的器件，数据和程序存储器使用不同的总线，因而可同时访问这两种存储器空间。

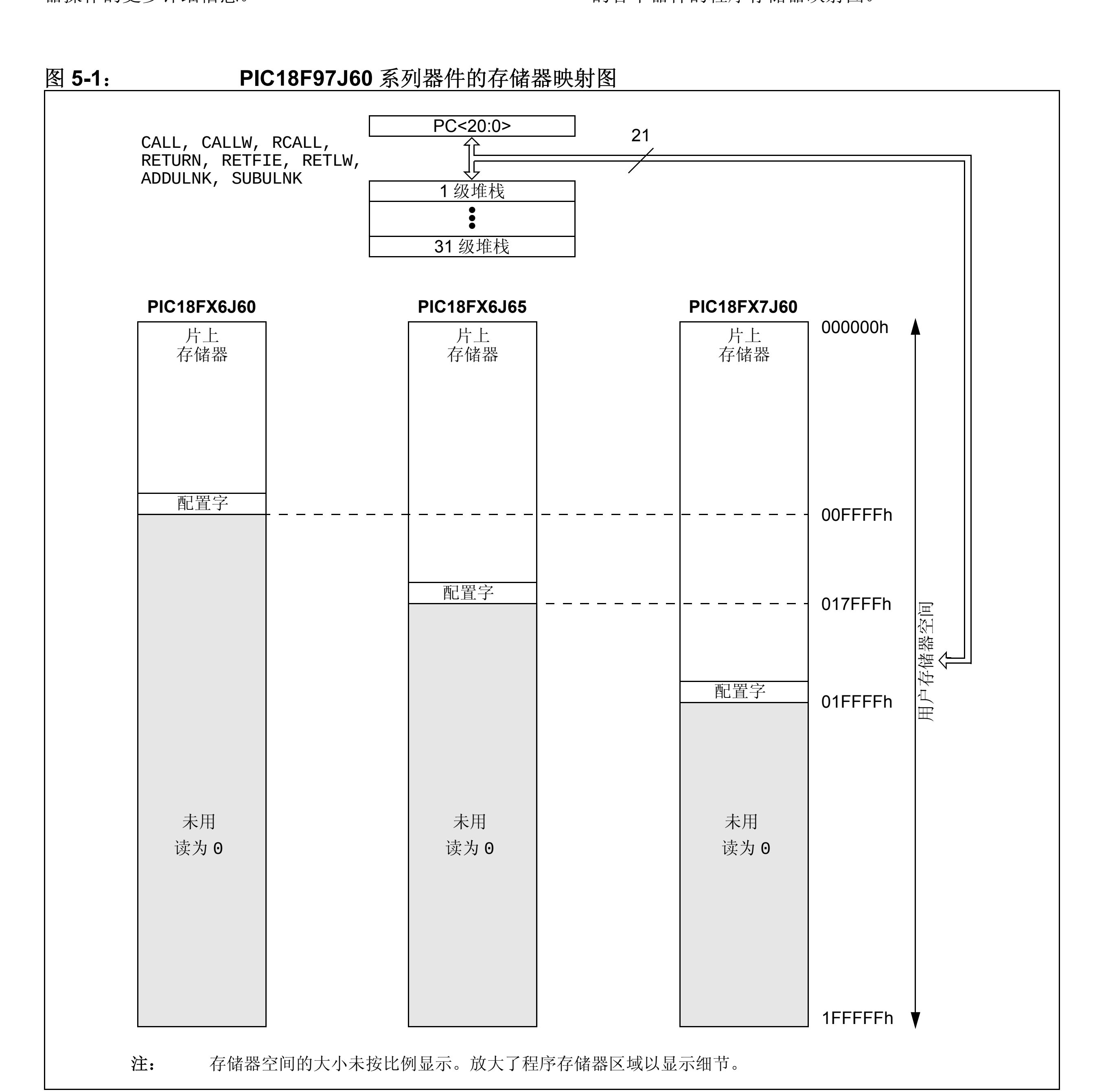
第 6.0 节 "闪存程序存储器"提供了关于闪存程序存储器操作的更多详细信息。

5.1 程序存储器构成

PIC18 单片机具有一个 21 位程序计数器，可以对 2 MB 的程序存储器空间进行寻址。访问存储器物理地址上边界和这个 2 MB 地址之间的存储单元会返回全 `0`（`NOP` 指令）。

整个 PIC18F97J60 系列提供了三种大小的片上闪存程序存储器，从 64 KB （至多 32,764 条单字指令）到 128 KB （65,532 条单字指令）。图 5-1 给出了该系列的各个器件的程序存储器映射图。

图 5-1： PIC18F97J60 系列器件的存储器映射图



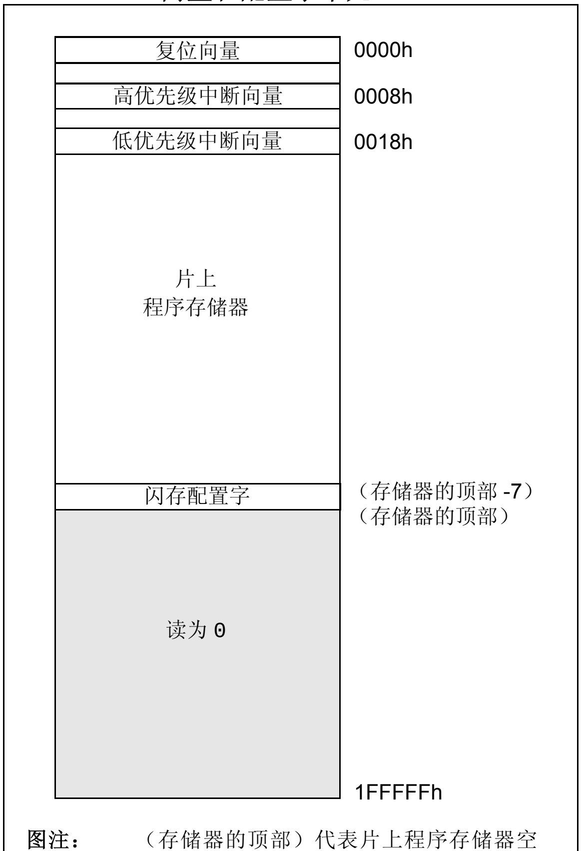
注： 存储器空间的大小未按比例显示。放大了程序存储器区域以显示细节。

5.1.1 存储器硬件编码向量

所有的 PIC18 器件在它们的程序存储器空间内共有 3 个硬件编码的返回向量。复位向量地址是在器件发生任何复位时程序计数器返回的默认值；它位于 0000h。

PIC18 器件还有两个中断向量地址，用于处理高优先级和低优先级中断。高优先级中断向量位于 0008h，低优先级中断向量位于 0018h。它们在程序存储器映射图中的相对位置如图 5-2 所示。

图 5-2： PIC18F97J60 系列器件的硬件编码向量和配置字单元



图注： （存储器的顶部）代表片上程序存储器空间的上边界（特定器件的具体值，请参见图 5-1）。阴影区域表示未实现的存储器。这些区域未按比例显示。

5.1.2 闪存配置字

由于 PIC18F97J60 系列器件没有固定的配置存储器，所以保留片上程序存储器顶部的 4 个字来保存配置信息。复位时，该配置信息被复制到配置寄存器。

配置字由低到高顺序存储在程序存储器单元中，从最低地址开始存放 CONFIG1 的低字节，到 CONFIG4 的高字节结束。对于这些器件，只使用从 CONFIG1 到 CONFIG3 的配置字，保留 CONFIG4。PIC18F97J60 系列器件的闪存配置字的实际地址如表 5-1所示。图5-2 给出了闪存配置字以及其他的存储器向量在存储器映射图中的位置。

第 **24.1** 节 **"配置位"**中提供了有关器件配置字的更多详细信息。

表 5-1： PIC18F97J60 系列器件的闪存配置字

器件	程序存储器（KB）	配置字地址
PIC18F66J60	64	FFF8h 到 FFFFh
PIC18F86J60		
PIC18F96J60		
PIC18F66J65	96	17FF8h 到 17FFFh
PIC18F86J65		
PIC18F96J65		
PIC18F67J60	128	1FFF8h 到 1FFFFh
PIC18F87J60		
PIC18F97J60		

 超前信息

5.1.3 PIC18F9XJ60/9XJ65 程序存储器模式

此系列中的 100 引脚器件可以对总共 2 MB 的程序存储器进行寻址。这是通过外部存储器总线实现的。控制器有两种不同的工作模式：

- 单片机 （MC）
- 扩展单片机 （EMC）

通过设置 EMB 配置位 （CONFIG3L<5:4>）决定程序存储器模式，如寄存器 5-1 所示。（关于器件配置位的更多详细信息，请参见**第 24.1 节 "配置位"**。）

程序存储器模式的工作方式如下：

- **单片机模式**只访问片上闪存存储器。尝试读片上存储器顶部以上的地址单元会导致读取值为全 0 （NOP 指令）。

  单片机模式也是 64 引脚和 80 引脚器件唯一可用的工作模式。

- **扩展单片机模式**允许将内部和外部程序存储器作为一个整体进行访问。器件可以访问其整个的片上程序存储器。除此之外，器件还可以访问外部程序存储器，寻址范围最大限制为 2 MB。程序可按照需要自动在两个存储器之间切换执行。

EMB 配置位的设置还能控制外部存储器的地址总线宽度。在**第 7.0 节 "外部存储器总线"**中对此有更详细的论述。

在这两种模式下，单片机都能访问数据 RAM。

图 5-3 比较了不同程序存储器模式的存储器映射图。表 5-2 中更充分地说明了片上存储器和外部存储器在访问限制方面的差异。

寄存器 5-1： CONFIG3L：配置寄存器 3 低字节

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT(1)	BW(1)	EMB1(1)	EMB0(1)	EASHFT(1)	—	—	—
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7 **WAIT**：外部总线等待使能位 (1)
1 = 禁止外部存储器总线上的操作的等待状态
0 = 使能外部存储器总线上的操作的等待状态并由 MEMCON<5:4> 选择

bit 6 **BW**：数据总线宽度选择位 (1)
1 = 16 位数据宽度模式
0 = 8 位数据宽度模式

bit 5-4 **EMB1:EMB0**：外部存储器总线配置位 (1)
11 = 单片机模式，禁止外部总线
10 = 扩展单片机模式， 12 位地址模式
01 = 扩展单片机模式， 16 位地址模式
00 = 扩展单片机模式， 20 位地址模式

bit 3 **EASHFT**：外部地址总线平移使能位 (1)
1 = 使能地址平移；外部总线上的地址是以 000000h 为起始地址的偏移值
0 = 禁止地址平移；外部总线上的地址反映 PC 值

bit 2-0 未用：读为 0

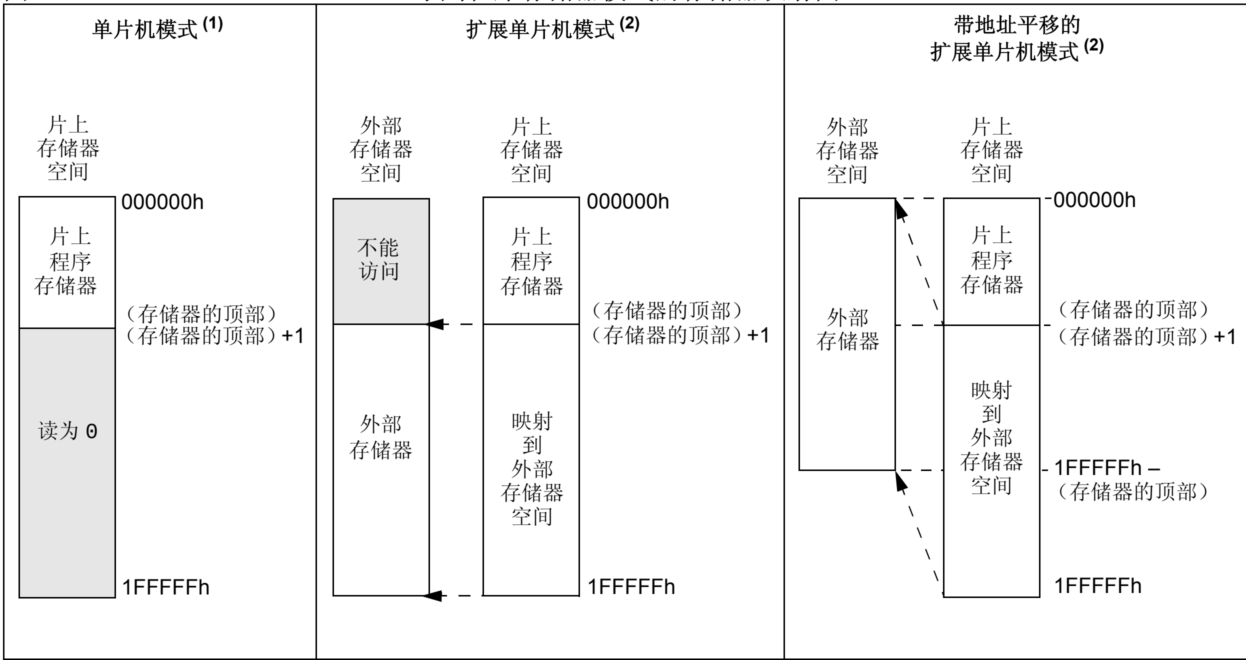
注 1： 仅在 100 引脚器件上实现。

5.1.4 扩展单片机模式和地址平移

默认情况下，处于扩展单片机模式的器件将指向外部存储器空间范围内的程序计数器值直接放到外部地址总线上。实际上，这意味着外部存储器中低于片上存储器顶部的地址都不能被访问。

为了避免这种情况，扩展单片机模式使用了一个地址平移功能以使能自动的地址转换。在此模式下，放在外部总线上的地址值减去片上程序存储器的大小并被重新映射到从 0000h 开始的地址处。这样就可以利用外部存储器的全部存储空间。

图 5-3： PIC18F97J60 系列程序存储器模式的存储器映射图



图注： （存储器的顶部）代表片上程序存储器空间的上边界（关于器件的具体值，请参见图 5-1）。阴影区域代表未实现或不能访问的区域，取决于不同模式。

注 1： 此模式是 64 引脚和 80 引脚器件唯一可用的模式，也是 100 引脚器件的默认模式。

2： 这些模式仅在 100 引脚器件上可用。

表 5-2： PIC18F9XJ60/9XJ65 程序存储器模式下的存储器访问

工作模式	内部程序存储器			外部程序存储器		
	程序执行	表读	表写	程序执行	表读	表写
单片机	可以	可以	可以	不可以	不可以	不可以
扩展单片机	可以	可以	可以	可以	可以	可以

5.1.5 程序计数器

程序计数器（Program Counter，PC）指定欲取出执行的指令的地址。PC 为 21 位宽，且保存在三个不同的 8 位寄存器中。存储低字节的寄存器称为 PCL 寄存器，该寄存器可读写。存储高字节的寄存器，即 PCH 寄存器，存储 PC<15:8> 位；该寄存器不可直接读写。更新 PCH 寄存器的操作是通过 PCLATH 寄存器实现的。存储最高字节的寄存器称为 PCU。该寄存器存储 PC<20:16> 位；它也不能直接读写。更新 PCU 寄存器的操作是通过 PCLATU 寄存器实现的。

PCLATH 和 PCLATU 的内容通过执行写 PCL 的操作被传送到程序计数器。同样，程序计数器的两个高字节通过读 PCL 的操作被传送到 PCLATH 和 PCLATU。这对于计算 PC 的偏移量很有用处（见第 **5.1.8.1 节 “计算 GOTO”**）。

PC 是按字节寻址程序存储器的。为了防止 PC 不能正确获取字指令，需要将 PCL 的最低有效位固定取值为 0。PC 每次加 2 来寻址程序存储器中的顺序指令。

CALL、RCALL、GOTO 和程序转移指令直接写入程序计数器。对于这些指令，PCLATH 和 PCLATU 的内容不会传送到程序计数器。

5.1.6 返回地址堆栈

返回地址堆栈允许保存最多 31 个程序调用地址和中断向量。当执行 CALL、RCALL 指令或响应中断时，PC 值会被压入该堆栈。当执行 RETURN，RETLW 或 RETFIE 指令（如果使能了扩展指令集，则还包括 ADDULNK 和SUBULNK 指令）时，PC 值会从堆栈弹出。PCLATU 和 PCLATH 不受 RETURN 或 CALL 指令的影响。

通过 21 位的 RAM 和一个 5 位的堆栈指针 STKPTR 来实现 31 字的堆栈操作。堆栈既不占用程序存储器空间，也不占用数据存储器空间。堆栈指针是可读写的，并且通过栈顶的特殊功能寄存器可以读写栈顶地址。也可以使用这些寄存器将数据压入堆栈或者从堆栈弹出。

执行 CALL 类型的指令时，执行进栈操作：首先堆栈指针加 1，并且将 PC 的内容写入堆栈指针所指向的地址单元（PC 已经指向 CALL 下一条指令）。执行 RETURN 类型的指令时，执行出栈操作：STKPTR 所指向的地址单元的内容会被传送给 PC，然后堆栈指针减 1。

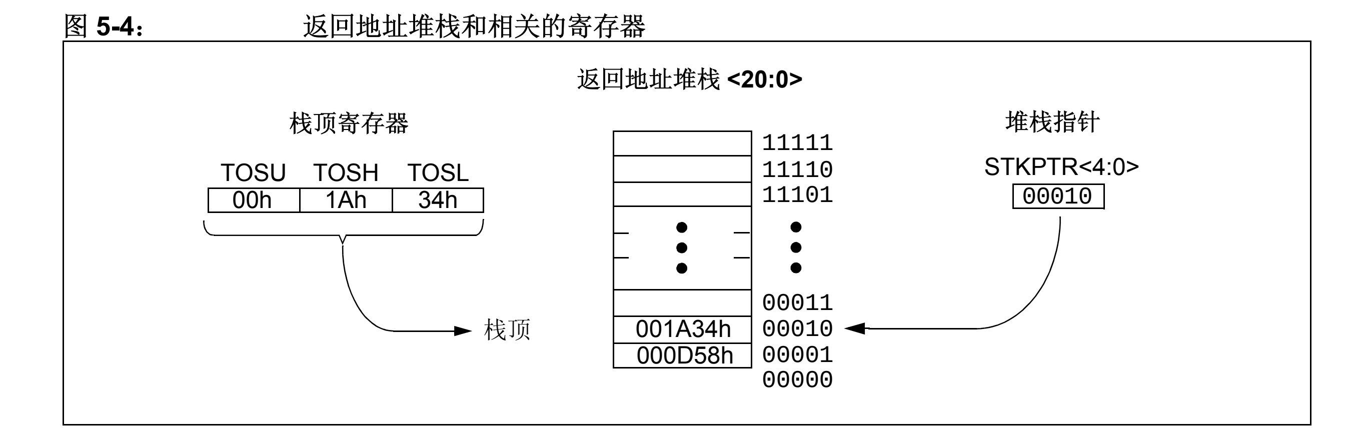
所有复位后，堆栈指针均会初始化为 00000。堆栈指针 00000 不指向任何 RAM 单元；它仅仅是一个复位值。状态位表明堆栈是已满、上溢还是下溢。

5.1.6.1 访问栈顶

只可读写返回地址堆栈的栈顶（Top-of-Stack，TOS）。有三个寄存器 TOSU:TOSH:TOSL 用于保存由 STKPTR 寄存器所指向的堆栈单元的内容（图 5-4）。这可以让用户在必要时实现软件堆栈。在 CALL、RCALL 或中断（如果使能了扩展指令集，则还包括ADDULNK和SUBULNK 指令）后，软件可以通过读取 TOSU:TOSH:TOSL 寄存器来读取压入堆栈的值。这些值可以被置入由用户定义的软件堆栈。返回时，软件将这些值存回 TOSU:TOSH:TOSL 并执行返回。

为防止意外的堆栈操作，访问堆栈时用户必须禁止全局中断允许位。

图 5-4： 返回地址堆栈和相关的寄存器



5.1.6.2 返回堆栈指针 （STKPTR）

STKPTR 寄存器（寄存器 5-2）包含堆栈指针值、STKFUL（堆栈满）状态位和 STKUNF（堆栈下溢）状态位。堆栈指针值可为 0 到 31 范围内的值。向堆栈压入值前，堆栈指针加 1 ；而从堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值为零。用户可以读写堆栈指针的值。实时操作系统（Real-Time Operating System，RTOS）可以利用此特性对返回堆栈进行维护。

向堆栈压入 PC 值 31 次 （且没有值从堆栈弹出）后，STKFUL 位置 1。通过软件或 POR 使 STKFUL 位清零。

由 STVREN （堆栈溢出复位使能）配置位的状态决定堆栈满时将执行的操作。（关于器件配置位的说明，请参见**第 24.1 节 “配置位”**。）如果 STVREN 位已经置 1（默认），第 31 次进栈将把 （PC + 2）值压入堆栈，从而将 STKFUL 位置 1 并复位器件。STKFUL 位将保持置 1，而堆栈指针将被清零。

如果 STVREN 已经清零，第 31 次进栈时 STKFUL 位将会置 1，堆栈指针递增到 31。后续的进栈操作都不会覆盖第 31 次进栈的值，并且 STKPTR 将保持为 31。

当出栈次数足够卸空堆栈时，下一次出栈操作会向 PC 返回一个零值，并将 STKUNF 位置 1，而堆栈指针则保持为零。STKUNF 位将保持置 1，直到软件清零或发生 POR 为止。

注：	下溢时，将零值返回给 PC，会使程序指向复位向量，此时可以验证堆栈状态并采取相应的操作。这与复位不同，因为下溢时 SFR 的内容不受影响。

5.1.6.3 PUSH 和 POP 指令

由于栈顶是可以读写的，因此将值压入堆栈或从堆栈弹出而不影响程序的正常执行是非常理想的。PIC18 指令集包含两个指令 PUSH 和 POP，使用这两个指令可在软件控制下对 TOS 执行操作。然后就可以修改 TOSU、TOSH 和 TOSL，将数据或返回地址压入堆栈。

PUSH 指令将当前的 PC 值压入堆栈。执行该指令会使堆栈指针加 1 并将当前的 PC 值装入堆栈。

POP 指令通过将堆栈指针减 1 来放弃当前的 TOS 值。然后前一个入栈的值就成为了 TOS 值。

寄存器 5-2： STKPTR：堆栈指针寄存器

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL[(1)]	STKUNF[(1)]	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **STKFUL：**堆栈满标志位 [(1)]
1 = 堆栈满或溢出
0 = 堆栈未满或未溢出

bit 6 **STKUNF：**堆栈下溢标志位 [(1)]
1 = 发生堆栈下溢
0 = 未发生堆栈下溢

bit 5 **未用：**读为 0

bit 4-0 **SP4:SP0：**堆栈指针地址位

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

5.1.6.4 堆栈满和下溢复位

通过将配置寄存器 1L 中的 STVREN 位置 1，来使能在堆栈溢出或下溢时的器件复位。当 STVREN 置 1 时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，然后使器件复位。当 STVREN 清零时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。只能通过用户软件或上电复位使 STKFUL 或 STKUNF 位清零。

5.1.7 快速寄存器堆栈

为 STATUS、WREG 和 BSR 寄存器提供的快速寄存器堆栈具有从中断“快速返回”的功能。此堆栈只有一级且不可读写。当处理器转入中断向量处执行时，它装入对应寄存器的当前值。所有中断源都会将值压入堆栈寄存器。如果使用 `RETFIE, FAST` 指令从中断返回，这些寄存器中的值就会被装回工作寄存器。

如果同时允许低优先级中断和高优先级中断，从低优先级中断返回时，无法可靠地使用堆栈寄存器。如果在为低优先级中断提供服务时，发生了高优先级中断，则低优先级中断存储在堆栈寄存器中的值将被覆盖。在为低优先级中断提供服务时，用户必须用软件保存关键寄存器的值。

如果未使用中断优先级，所有中断都可以使用快速寄存器堆栈从中断返回。如果没有使用中断，快速寄存器堆栈可以用于在子程序调用结束后恢复 STATUS、WREG 和 BSR 寄存器。要在子程序调用中使用快速寄存器堆栈，必须执行 `CALL label, FAST` 指令，将 STATUS、WREG 和 BSR 寄存器的内容存入快速寄存器堆栈。然后执行 `RETURN, FAST` 指令，从快速寄存器堆栈恢复这些寄存器。

例 5-1 给出了一个在子程序调用和返回期间使用快速寄存器堆栈的源代码示例。

例 5-1： 快速寄存器堆栈代码示例

```
CALL  SUB1, FAST     ;STATUS, WREG, BSR
                     ;SAVED IN FAST REGISTER
                     ;STACK
      •
      •

SUB1  •
      •
      RETURN FAST    ;RESTORE VALUES SAVED
                     ;IN FAST REGISTER STACK
```

5.1.8 程序存储器中的查找表

有的编程需要在程序存储器中创建数据结构或查找表。对于 PIC18 器件，可以用两种方式实现查找表：

- 计算 `GOTO`
- 表读

5.1.8.1 计算 GOTO

计算 `GOTO` 是通过向程序计数器加一个偏移量来实现的。例 5-2 给出了一个示例。

使用 `ADDWF PCL` 指令和一组 `RETLW nn` 指令可以创建一个查找表。在调用该表前，会先将查找表中的偏移量装入 W 寄存器。被调用子程序的第一条指令应该是 `ADDWF PCL` 指令。接下去执行的一条是 `RETLW nn` 指令，它将值 nn 返回给调用函数。

偏移量（WREG 中）指定程序计数器应该增加的字节数，其值应该为 2 的倍数（LSb = 0）。

在这种方式中，每个指令单元只能存储一个数据字节，并且要求返回地址堆栈还有空闲单元。

例 5-2： 使用偏移量计算 GOTO

```
        MOVF    OFFSET, W
        CALL    TABLE
ORG     nn00h
TABLE   ADDWF   PCL
        RETLW   nnh
        RETLW   nnh
        RETLW   nnh
        .
        .
        .
```

5.1.8.2 表读

有一种更好的方法可以将数据存储在程序存储器中，这种方法允许在每个指令单元存储 2 个字节的数据。

编程时，每个程序字可以存储 2 个字节的查找表数据。表指针（TBLPTR）指定字节地址，而表锁存器（TABLAT）则存储从程序存储器读取的数据。一次只能从程序存储器读取一个字节。

在**第 6.1 节“表读与表写”**中进一步讨论表读操作。

5.2 PIC18 指令周期

5.2.1 时钟分配

来自内部或外部时钟源的单片机时钟输入都将在内部被四分频以产生四个互不重叠的正交时钟信号（Q1、Q2、Q3 和 Q4）。程序计数器在每个 Q1 递增；在 Q4 期间，从程序存储器取指令并将指令锁存到指令寄存器 （IR）中。指令的译码和执行在下一个 Q1 到 Q4 周期完成。图 5-5 所示为时钟和指令执行的流程图。

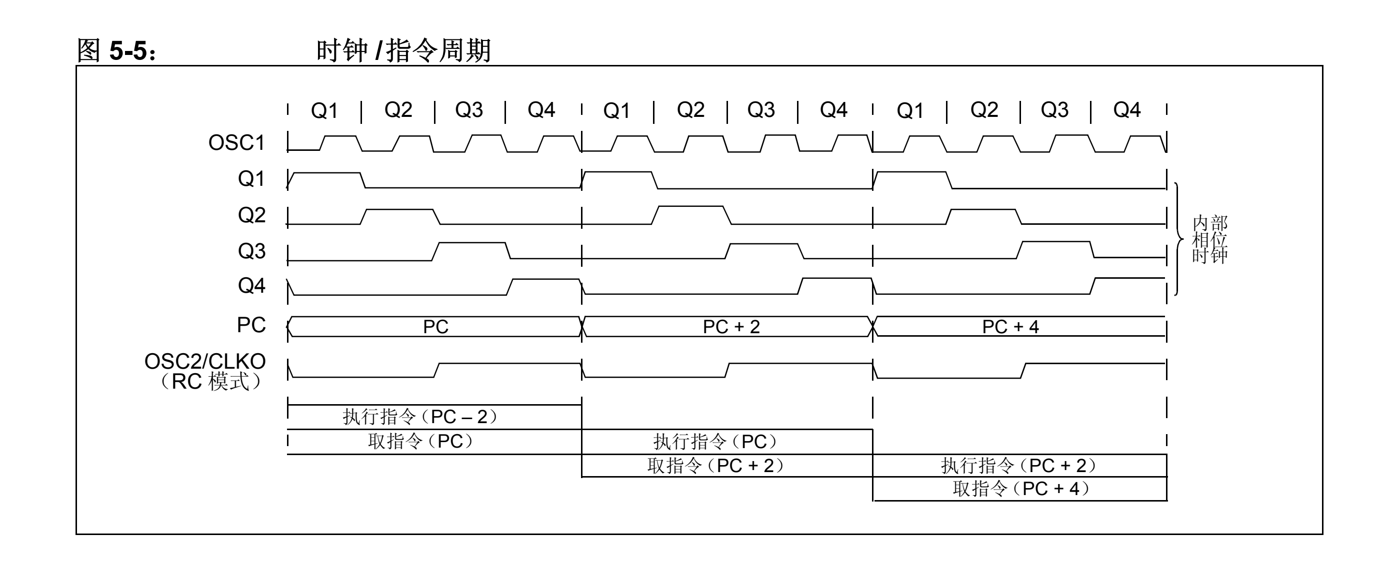
5.2.2 指令流 / 流水线

一个指令周期由 Q1 到 Q4 四个 Q 周期组成。取指令和执行指令是以流水线方式进行的，用一个指令周期来取指令，而用另一个指令周期译码和执行指令。但由于是流水线操作，所以每条指令的等效执行时间都是一个指令周期。如果某条指令改变了程序计数器 （如 GOTO），则需要两个指令周期才能完成该指令 （例 5-3）。

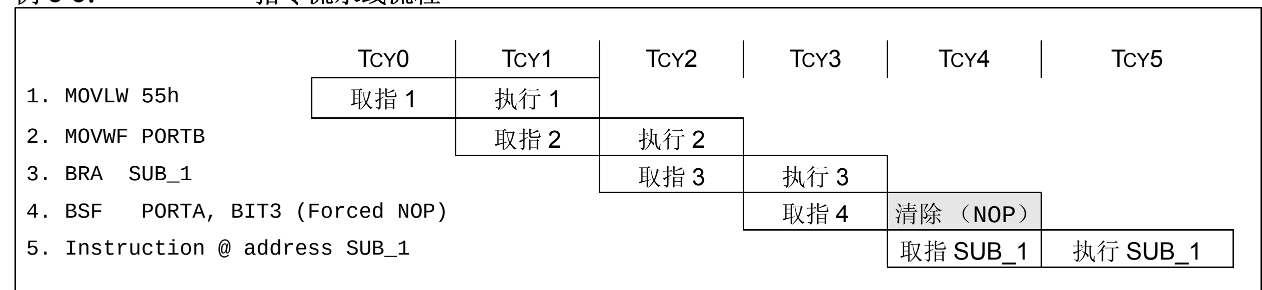
取指周期中：程序计数器 （PC）在 Q1 周期加 1，开始取指令。

指令执行周期中：在 Q1 周期，将所取指令锁存到指令寄存器 （IR）。在随后的 Q2、 Q3 和 Q4 周期中译码并执行该指令。其中读数据存储器（读操作数）发生在 Q2 周期，写操作发生在 Q4 周期 （写目标寄存器）。

图 5-5： 时钟 / 指令周期



例 5-3： 指令流水线流程



	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	取指 1	执行 1				
2. MOVWF PORTB		取指 2	执行 2			
3. BRA SUB_1			取指 3	执行 3		
4. BSF PORTA, BIT3 (Forced NOP)				取指 4	清除 （NOP）	
5. Instruction @ address SUB_1					取指 SUB_1	执行 SUB_1

除程序转移指令外，所有的指令都是单周期指令。由于程序转移指令将导致一条已取指令从流水线清除，需要重新取指令，然后执行指令，所以程序转移指令需要两个周期。

5.2.3 程序存储器中的指令

程序存储器按字节寻址。指令以 2 字节或 4 字节的形式存储在程序存储器中。指令字的最低有效字节始终存储在地址为偶数的程序存储器单元中（LSb = 0）。要保证正确指向指令单元，PC 必须以 2 为单位递增，并且 LSb 总是读为 0（见第 **5.1.5 节 “程序计数器”**）。

图 5-6 给出了指令字存储在程序存储器中的一个示例。

CALL 和 GOTO 指令在指令中嵌入了程序存储器的绝对地址。由于指令总是存储为一个字长，因而指令所包含的数据为一个字地址。字地址会写入 PC<20:1>，用于访问程序存储器中的目标字节。图 5-6 中的指令 #2 给出了指令 GOTO 0006h 在程序存储器中的译码过程。程序转移指令也采取同样的方式对相对地址偏移量进行译码。存储在转移指令中的偏移量代表单字指令数，PC 将以此作为偏移量跳转到指定的地址单元。**第 25.0 节 “指令集汇总”**提供了指令集的更多详细信息。

图 5-6： 程序存储器中的指令

			LSB = 1	LSB = 0	字地址 ↓
	程序存储器字节单元 →				000000h
					000002h
					000004h
					000006h
指令 1：	MOVLW	055h	0Fh	55h	000008h
指令 2：	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
指令 3：	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

5.2.4 双字指令

标准的 PIC18 指令集有 4 条双字指令：CALL、 MOVFF、GOTO 和 LFSR。这些指令第二个字的高 4 位均为 1111；其他 12 位是立即数数据，通常为一个数据存储器地址。

指令的高 4 位为 1111，用于代表一条特殊的 NOP 指令。指令顺序执行的正确顺序为：执行完第一个字之后立即按顺序访问并使用第二个字中的数据。如果由于某些原因跳过了第一个字而自动执行指令的第二个字，那么将作为一条 NOP 指令执行。如果双字指令跟在修改 PC 的条件指令后，就有必要执行此操作。例 5-4 给出了它的执行过程。

> **注：** 关于扩展指令集中的双字指令信息，请参见第 **5.5 节 “程序存储器和扩展指令集”**。

例 5-4： 双字指令

情形 1：

目标代码	源代码		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code

情形 2：

目标代码	源代码		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

5.3 数据存储器构成

> 注： 当使能了 PIC18 扩展指令集时，数据存储器某些方面的操作会有所改变。更多信息，请参见第 **5.6 节 “数据存储器和扩展指令集”**。

PIC18 器件中的数据存储器是用静态 RAM 实现的。在数据存储器中，每个寄存器有 12 位地址，数据存储器可达 4096 个字节。存储器空间被分为 16 个存储区，每个存储区包含 256 个字节。PIC18F97J60 系列器件实现了所有可用的存储区，并提供 3808 字节的数据存储区供用户使用。图 5-7 给出了器件的数据存储器构成。

数据存储器由特殊功能寄存器（Special Function Register， SFR）和通用寄存器（General Purpose Register，GPR）组成。SFR 用于单片机和外设功能模块的控制和状态显示，GPR 则在用户应用程序中用于存储数据和临时存储操作的中间结果。任何未实现单元均读为 `0`。

这样的指令集和架构支持跨存储区的操作。可以通过直接、间接或变址寻址模式访问整个数据存储器。本章后面的部分将讨论寻址模式。

为确保能在一个周期存取常用寄存器（大多数 SFR 和所选的 GPR）， PIC18 器件设置了一个快速操作存储区。该存取区是一个 256 字节的存储器空间，它可实现对大多数 SFR 和 GPR Bank 0 的低地址单元的快速存取，而无需使用 BSR。**第 5.3.2 节 “快速操作存储区”** 提供了对于快速操作 RAM 的详细说明。

5.3.1 存储区选择寄存器

容量较大的数据存储器需要有效的寻址机制，以便对所有地址进行快速存取。理想状况下，这意味着不必为每次读写操作提供完整地址。 PIC18 器件是使用 RAM 存储区机制实现快速存取的。这种机制将存储器空间分成连续的 16 个 256 字节的存储区。根据不同的指令，可以通过完整的 12 位地址直接寻址每个单元，或通过 8 位的低字节地址和 4 位存储区指针寻址每个单元。

PIC18 指令集中的大部分指令都使用存储区指针，也就是存储区选择寄存器（Bank Select Register，BSR）。SFR 保存单元地址的高 4 位，而指令本身则包括单元地址的低 8 位。只使用 BSR 的低 4 位（BSR3:BSR0），不使用高 4 位；它们始终读为 `0` 且不能被写入。可以通过使用 `MOVLB` 指令直接装载 BSR。

BSR 的值代表数据存储器中的存储区，指令中的 8 位指向存储区中的存储单元，可以将它看作距离存储区下边界的偏移量。图 5-8 显示了 BSR 的值与数据存储器中的存储区之间的关系。

由于最多可有 16 个寄存器共享同一个低位地址，用户必须非常小心以确保在执行数据读或写之前选择了正确的存储区。例如，当 BSR 为 0Fh 时将程序数据写入地址为 F9h 的 8 位地址单元，将导致程序计数器重新赋值。

当选择存储区时，只有已实现的存储区才可以读写。对未实现存储区进行的写操作将被忽略，而读这些存储区会返回 `0`。虽然是这样， STATUS 寄存器仍然会受到影响，好像操作是成功的。图 5-7 中的数据存储器映射图指出了已实现的存储区。

在 PIC18 的内核指令集中，只有 `MOVFF` 指令指定源寄存器和目标寄存器的完整 12 位地址。该指令在执行时完全忽略 BSR。所有其他指令仅包含作为操作数的低位地址，而且必须使用 BSR 或快速操作存储区来寻址目标寄存器。

图 5-7： PIC18F97J60 系列器件的数据存储器映射图

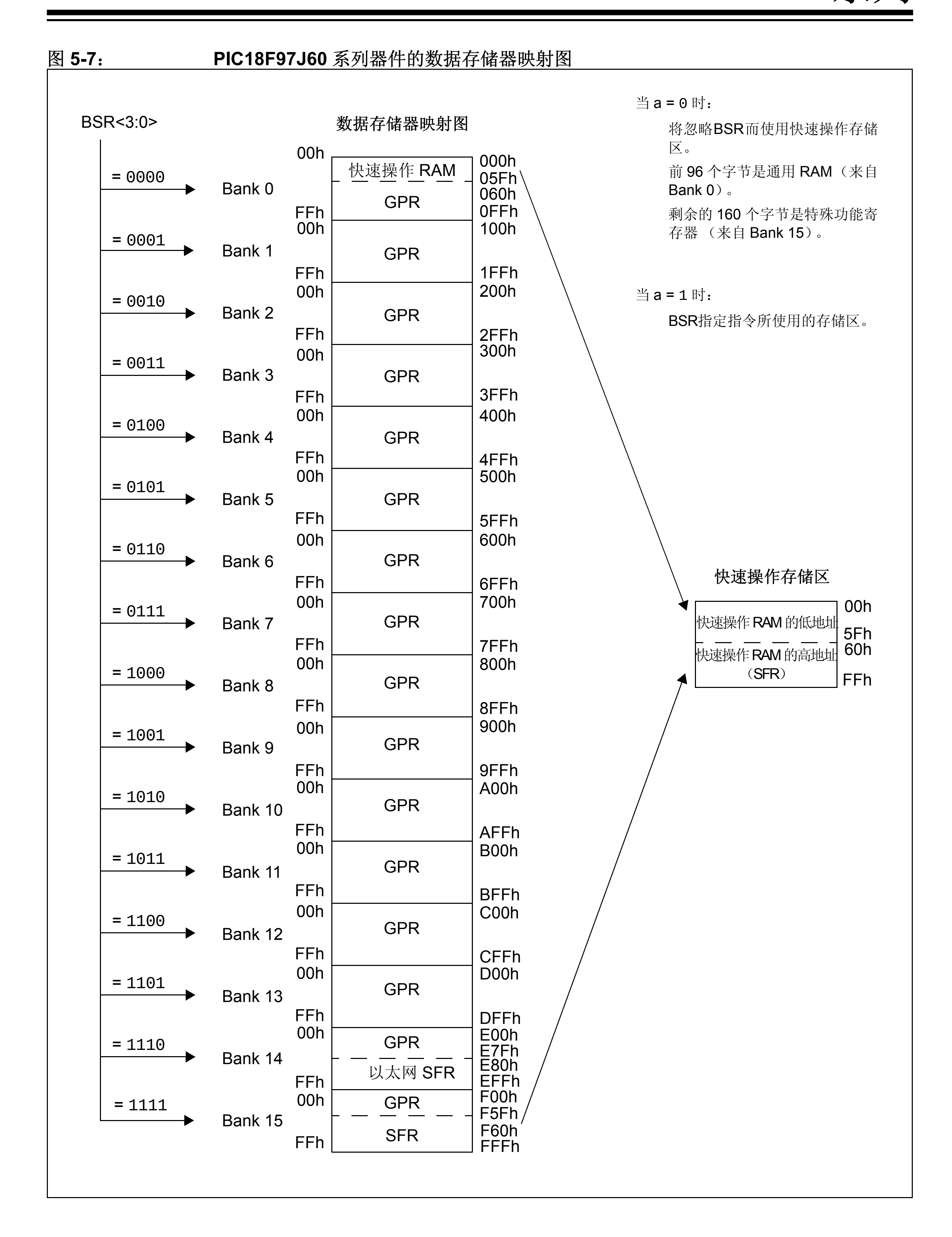
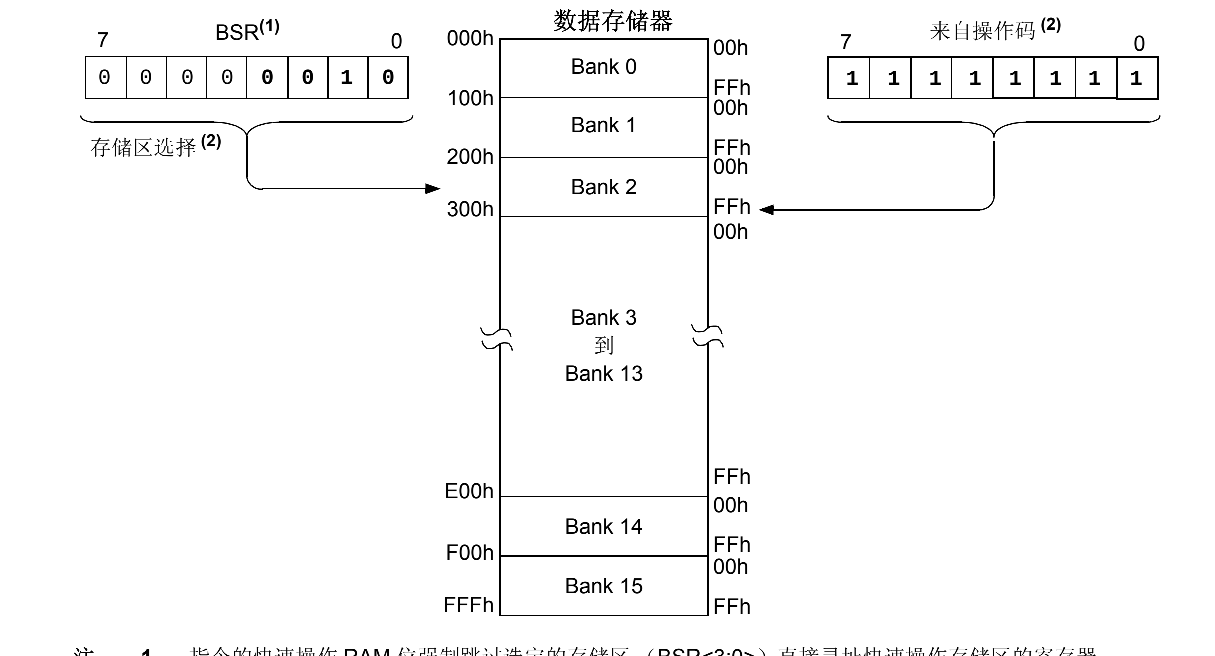


图 5-8： 使用存储区选择寄存器 （直接寻址）



注 1： 指令的快速操作 RAM 位强制跳过选定的存储区 （BSR<3:0>）直接寻址快速操作存储区的寄存器。
2： MOVFF 指令嵌入了完整的 12 位地址。

5.3.2 快速操作存储区

使用 BSR 和嵌入的 8 位地址，用户可以寻址数据存储器的整个空间，但这同时也意味着用户必须始终确保选择了正确的存储区。否则，可能会从错误的单元读取数据或将数据写入错误的单元。如果本来是向 GPR 进行写操作，却将结果写入了 SFR，后果是非常严重的。但是在每次向数据存储器进行读或写操作时确认和 / 或更改 BSR 会严重影响工作效率。

为了连续访问大多数常用数据存储器单元，现为数据存储器配置了快速操作存储区，这样可以允许用户访问被映射的存储区而无需指定 BSR。快速操作存储区由 Bank 0 的前 96 个字节（00h-5Fh）和 Bank 15 的后 160 个字节 （60h-FFh）组成。地址较低的部分被称为 “快速操作 RAM”，由 GPR 组成。地址较高的部分则被映射为器件的 SFR。这两个区域被连续地映射到快速操作存储区并且可以用一个8位地址进行线性寻址（图5-7）。

包括快速操作 RAM 位（指令中的“a”参数）的 PIC18 内核指令在执行时使用快速操作存储区。当 “a”等于 1 时，指令使用 BSR 和包含在操作码中的 8 位地址对数据存储器寻址。当 “a”为 0 时，强制指令使用快速操作存储区地址映射，此时忽略 BSR 的当前值。

此 “强制”寻址方式可使指令在一个周期内对数据地址进行操作，而不需要首先更新 BSR。这意味着用户可以更有效地对 8 位地址为 60h 或以上的 SFR 进行取值和操作。地址为 60h 以下的快速操作 RAM 非常适合于存储那些用户可能需要快速访问的数据值，如直接计算结果或常用程序变量。快速操作 RAM 也可实现更加快速和有效的现场保护和变量切换。

使能扩展的指令集（XINST 配置位 = 1）时的快速操作存储区的映射略有不同。在**第 5.6.3 节 “在立即数变址模式中映射快速操作存储区”**中对此进行了更详细的讨论。

5.3.3 通用寄存器文件

PIC18 器件可能在 GPR 区中划分了一部分存储区。这部分存储区为数据 RAM，所有指令均可访问它。 GPR 区从 Bank 0 的底部 （地址 000h）开始向上直到 SFR 区的底部。上电复位不会初始化 GPR，并且其他复位也不会改变其内容。

5.3.4 特殊功能寄存器

特殊功能寄存器（SFR）是 CPU 和外设模块用来控制器件操作的寄存器。这些寄存器以静态 RAM 的形式实现。

SFR 从数据存储器的顶部（FFFh）开始向下，它占用了 Bank 15 上半部分的单元空间（F60h 到 FFFh）。可以将 SFR 归类为两组：与“内核”器件功能（ALU、复位和中断）相关的寄存器和与外设功能相关的寄存器。复位和中断寄存器在相关的章节中进行讨论，本章后面的部分将对 ALU 状态寄存器进行说明。与外设操作相关的寄存器将在该外设的章节中进行说明。

SFR 通常分布在受其控制的外设中。未使用的 SFR 单元是未实现的，读为 0。表 5-3 给出了 SFR 的列表；表 5-5 中提供了完整说明。

表 5-3： PIC18F97J60 系列器件的特殊功能寄存器映射图

地址	名称	地址	名称	地址	名称	地址	名称	地址	名称
FFFh	TOSU	FDFh	INDF2[(1)]	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	SPBRGH1
FFEh	TOSH	FDEh	POSTINC2[(1)]	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	BAUDCON1
FFDh	TOSL	FDDh	POSTDEC2[(1)]	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	SPBRGH2
FFCh	STKPTR	FDCh	PREINC2[(1)]	FBCh	CCPR2H	F9Ch	MEMCON[(3)]	F7Ch	BAUDCON2
FFBh	PCLATU	FDBh	PLUSW2[(1)]	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	ERDPTH
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ[(3)]	F7Ah	ERDPTL
FF9h	PCL	FD9h	FSR2L	FB9h	CCPR3H	F99h	TRISH[(3)]	F79h	ECCP1DEL
FF8h	TBLPTRU	FD8h	STATUS	FB8h	CCPR3L	F98h	TRISG	F78h	TMR4
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	CCP3CON	F97h	TRISF	F77h	PR4
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE	F76h	T4CON
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	CCPR4H
FF4h	PRODH	FD4h	—[(2)]	FB4h	CMCON	F94h	TRISC	F74h	CCPR4L
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	CCP4CON
FF2h	INTCON	FD2h	ECON1	FB2h	TMR3L	F92h	TRISA	F72h	CCPR5H
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ[(3)]	F71h	CCPR5L
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH[(3)]	F70h	CCP5CON
FEFh	INDF0[(1)]	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	SPBRG2
FEEh	POSTINC0[(1)]	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	RCREG2
FEDh	POSTDEC0[(1)]	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	TXREG2
FECh	PREINC0[(1)]	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	TXSTA2
FEBh	PLUSW0[(1)]	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	RCSTA2
FEAh	FSR0H	FCAh	T2CON	FAAh	—[(2)]	F8Ah	LATB	F6Ah	ECCP3AS
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	—[(2)]	F89h	LATA	F69h	ECCP3DEL
FE8h	WREG	FC8h	SSP1ADD	FA8h	—[(2)]	F88h	PORTJ[(3)]	F68h	ECCP2AS
FE7h	INDF1[(1)]	FC7h	SSP1STAT	FA7h	EECON2[(1)]	F87h	PORTH[(3)]	F67h	ECCP2DEL
FE6h	POSTINC1[(1)]	FC6h	SSP1CON1	FA6h	EECON1	F86h	PORTG	F66h	SSP2BUF
FE5h	POSTDEC1[(1)]	FC5h	SSP1CON2	FA5h	IPR3	F85h	PORTF	F65h	SSP2ADD
FE4h	PREINC1[(1)]	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	SSP2STAT
FE3h	PLUSW1[(1)]	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	SSP2CON1
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	SSP2CON2
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	EDATA
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	EIR

注 1： 这不是实际存在的寄存器。
2： 未实现的寄存器，读为 0。
3： 在 64 引脚的器件上寄存器不可用。

5.3.5 以太网 SFR

除了 Bank 15 中的标准 SFR， PIC18F97J60 系列器件还有另外一组SFR。后者专用于以态网模块，占用Bank 14 的上半部分 （E80h 到 EFFh）。

注：	要改善性能，经常访问的以太网寄存器应位于标准 SFR 存储区 （F60h 到 FFFh）。

表 5-4 给出了以太网 SFR 的完整列表。表 5-5 对所有 SFR 进行了完整的说明。

表 5-4： PIC18F97J60 系列器件的以太网 SFR 映射图

地址	名称	地址	名称	地址	名称	地址	名称
EFFh	—(1)	EDFh	—(1)	EBFh	—(1)	E9Fh	—(1)
EFEh	ECON2	EDEh	—(1)	EBEh	—(1)	E9Eh	—(1)
EFDh	ESTAT	EDDh	—(1)	EBDh	—(1)	E9Dh	—(1)
EFCh	—(1)	EDCh	—(1)	EBCh	—(1)	E9Ch	—(1)
EFBh	EIE	EDBh	—(1)	EBBh	—(1)	E9Bh	—(1)
EFAh	—(1)	EDAh	—(1)	EBAh	—(1)	E9Ah	—(1)
EF9h	—(2)	ED9h	EPKTCNT	EB9h	MIRDH	E99h	EPAUSH
EF8h	—(2)	ED8h	ERXFCON	EB8h	MIRDL	E98h	EPAUSL
EF7h	EDMACSH	ED7h	—(1)	EB7h	MIWRH	E97h	EFLOCON
EF6h	EDMACSL	ED6h	—(1)	EB6h	MIWRL	E96h	—(2)
EF5h	EDMADSTH	ED5h	EPMOH	EB5h	—(1)	E95h	—(2)
EF4h	EDMADSTL	ED4h	EPMOL	EB4h	MIREGADR	E94h	—(2)
EF3h	EDMANDH	ED3h	—(2)	EB3h	—(2)	E93h	—(2)
EF2h	EDMANDL	ED2h	—(2)	EB2h	MICMD	E92h	—(2)
EF1h	EDMASTH	ED1h	EPMCSH	EB1h	MICON	E91h	—(2)
EF0h	EDMASTL	ED0h	EPMCSL	EB0h	—(1)	E90h	—(2)
EEFh	ERXWRPTH	ECFh	EPMM7	EAFh	—(2)	E8Fh	—(2)
EEEh	ERXWRPTL	ECEh	EPMM6	EAEh	—(1)	E8Eh	—(2)
EEDh	ERXRDPTH	ECDh	EPMM5	EADh	—(1)	E8Dh	—(2)
EECh	ERXRDPTL	ECCh	EPMM4	EACh	—(1)	E8Ch	—(2)
EEBh	ERXNDH	ECBh	EPMM3	EABh	MAMXFLH	E8Bh	—(2)
EEAh	ERXNDL	ECAh	EPMM2	EAAh	MAMXFLL	E8Ah	MISTAT
EE9h	ERXSTH	EC9h	EPMM1	EA9h	MACLCON2	E89h	—(1)
EE8h	ERXSTL	EC8h	EPMM0	EA8h	MACLCON1	E88h	—(1)
EE7h	ETXNDH	EC7h	EHT7	EA7h	MAIPGH	E87h	—(1)
EE6h	ETXNDL	EC6h	EHT6	EA6h	MAIPGL	E86h	—(1)
EE5h	ETXSTH	EC5h	EHT5	EA5h	—(2)	E85h	MAADR2
EE4h	ETXSTL	EC4h	EHT4	EA4h	MABBIPG	E84h	MAADR1
EE3h	EWRPTH	EC3h	EHT3	EA3h	MACON4	E83h	MAADR4
EE2h	EWRPTL	EC2h	EHT2	EA2h	MACON3	E82h	MAADR3
EE1h	—(1)	EC1h	EHT1	EA1h	—(1)	E81h	MAADR6
EE0h	—(1)	EC0h	EHT0	EA0h	MACON1	E80h	MAADR5

注 1： 保留的寄存器单元，不能修改。
2： 未实现的寄存器，读为 0。

 超前信息

表 5-5： 寄存器文件汇总 （PIC18F97J60 系列）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见（页）：
TOSU	—	—	—	栈顶寄存器最高字节 （TOS<20:16>）					---0 0000	59, 71
TOSH	栈顶寄存器高字节 （TOS<15:8>）								0000 0000	59, 71
TOSL	栈顶寄存器低字节 （TOS<7:0>）								0000 0000	59, 71
STKPTR	STKFUL[1]	STKUNF[1]	—	SP4	SP3	SP2	SP1	SP0	00-0 0000	59, 72
PCLATU	—	—	bit 21[2]	PC<20:16> 的保持寄存器					---0 0000	59, 71
PCLATH	PC<15:8> 的保持寄存器								0000 0000	59, 71
PCL	PC 低字节 （PC<7:0>）								0000 0000	59, 71
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					--00 0000	59, 98
TBLPTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								0000 0000	59, 98
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								0000 0000	59, 98
TABLAT	程序存储器表锁存器								0000 0000	59, 98
PRODH	乘积寄存器高字节								xxxx xxxx	59, 117
PRODL	乘积寄存器低字节								xxxx xxxx	59, 117
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	59, 121
INTCON2	$\overline{RBPU}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	59, 122
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	59, 123
INDF0	使用 FSR0 的内容寻址数据存储器——FSR0 的值不变 （不是实际存在的寄存器）								N/A	59, 89
POSTINC0	使用 FSR0 的内容寻址数据存储器——FSR0 的值后增 （不是实际存在的寄存器）								N/A	59, 90
POSTDEC0	使用 FSR0 的内容寻址数据存储器——FSR0 的值后减 （不是实际存在的寄存器）								N/A	59, 90
PREINC0	使用 FSR0 的内容寻址数据存储器——FSR0 的值预增 （不是实际存在的寄存器）								N/A	59, 90
PLUSW0	使用 FSR0 的内容寻址数据存储器——FSR0 的值预增 （不是实际存在的寄存器）， FSR0 的偏移量由 W 寄存器提供								N/A	59, 90
FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				---- xxxx	59, 89
FSR0L	间接数据存储器地址指针 0 的低字节								xxxx xxxx	59, 90
WREG	工作寄存器								xxxx xxxx	59
INDF1	使用 FSR1 的内容寻址数据存储器——FSR1 的值不变 （不是实际存在的寄存器）								N/A	59, 89
POSTINC1	使用 FSR1 的内容寻址数据存储器——FSR1 的值后增 （不是实际存在的寄存器）								N/A	59, 90
POSTDEC1	使用 FSR1 的内容寻址数据存储器——FSR1 的值后减 （不是实际存在的寄存器）								N/A	59, 90
PREINC1	使用 FSR1 的内容寻址数据存储器——FSR1 的值预增 （不是实际存在的寄存器）								N/A	59, 90
PLUSW1	使用 FSR1 的内容寻址数据存储器——FSR1 的值预增 （不是实际存在的寄存器）， FSR1 的偏移量由 W 寄存器提供								N/A	59, 90
FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				---- xxxx	59, 89
FSR1L	间接数据存储器地址指针 1 的低字节								xxxx xxxx	59, 89
BSR	—	—	—	—	存储区选择寄存器				---- 0000	59, 89
INDF2	使用 FSR2 的内容寻址数据存储器——FSR2 的值不变 （不是实际存在的寄存器）								N/A	59, 89
POSTINC2	使用 FSR2 的内容寻址数据存储器——FSR2 的值后增 （不是实际存在的寄存器）								N/A	59, 90
POSTDEC2	使用 FSR2 的内容寻址数据存储器——FSR2 的值后减 （不是实际存在的寄存器）								N/A	59, 90
PREINC2	使用 FSR2 的内容寻址数据存储器——FSR2 的值预增 （不是实际存在的寄存器）								N/A	59, 90
PLUSW2	使用 FSR2 的内容寻址数据存储器——FSR2 的值预增 （不是实际存在的寄存器）， FSR2 的偏移量由 W 寄存器提供								N/A	59, 90
FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				---- xxxx	59, 89
FSR2L	间接数据存储器地址指针 2 的低字节								xxxx xxxx	59, 89

图注： x = 未知， u = 不变， - = 未用 （读为 0）， q = 值取决于具体条件， r = 保留位 （不能修改）。阴影单元未实现，读为 0。

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

2： PC 的 bit 21 仅在串行编程模式下可用。

3： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

4： 当 MSSP 模块工作在 I^2C™ 从动模式下时，这些位的其他名称和定义。

5： 这些位和 / 或寄存器仅在 100 引脚器件上可用；在其他器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

6： 这些位和 / 或寄存器仅在 80 引脚和 100 引脚器件上可用；在 64 引脚器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

7： 在单片机模式下，此寄存器中的位不可写且读为 0。

8： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

9： 仅在单片机模式下的 100 引脚器件上实现。

表 5-5： 寄存器文件汇总 （PIC18F97J60 系列） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见 （页）：
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	60, 87
TMR0H	Timer0 寄存器的高字节								0000 0000	60, 163
TMR0L	Timer0 寄存器的低字节								xxxx xxxx	60, 163
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	60, 163
OSCCON	IDLEN	—	—	—	OSTS(3)	—	SCS1	SCS0	0--- q-00	60, 43
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	0000 00--	60, 211
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	60, 353
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 1100	60, 54, 133
TMR1H	Timer1 寄存器的高字节								xxxx xxxx	60, 167
TMR1L	Timer1 寄存器的低字节								xxxx xxxx	60, 167
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	60, 167
TMR2	Timer2 寄存器								0000 0000	60, 173
PR2	Timer2 周期寄存器								1111 1111	60, 173
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	60, 173
SSP1BUF	MSSP1 接收缓冲区 / 发送寄存器								xxxx xxxx	60, 265
SSP1ADD	MSSP1 地址寄存器 （I2C™ 从动模式）， MSSP1 波特率重载寄存器 （I2C 主控模式）								0000 0000	60, 265
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	60, 256, 266
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	60, 257, 267
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	60, 268
	GCEN	ACKSTAT	ADMSK5(4)	ADMSK4(4)	ADMSK3(4)	ADMSK2(4)	ADMSK1(4)	SEN		
ADRESH	A/D 结果寄存器的高字节								xxxx xxxx	60, 333
ADRESL	A/D 结果寄存器的低字节								xxxx xxxx	60, 333
ADCON0	ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	0-00 0000	60, 325
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	60, 326
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	60, 327
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								xxxx xxxx	60, 185
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								xxxx xxxx	60, 185
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	60, 189
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								xxxx xxxx	60, 185
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								xxxx xxxx	60, 185
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000 0000	60, 189
CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节								xxxx xxxx	60, 185
CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节								xxxx xxxx	60, 185
CCP3CON	P3M1	P3M0	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	0000 0000	60, 189
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	0000 0000	60, 201
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	60, 341
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	60, 335
TMR3H	Timer3 寄存器的高字节								xxxx xxxx	60, 175
TMR3L	Timer3 寄存器的低字节								xxxx xxxx	60, 175

图注： x = 未知， u = 不变， - = 未用 （读为 0）， q = 值取决于具体条件， r = 保留位 （不能修改）。阴影单元未实现，读为 0。

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。
2： PC 的 bit 21 仅在串行编程模式下可用。
3： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。
4： 当 MSSP 模块工作在 I2C™ 从动模式下时，这些位的其他名称和定义。
5： 这些位和 / 或寄存器仅在 100 引脚器件上可用；在其他器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。
6： 这些位和 / 或寄存器仅在 80 引脚和 100 引脚器件上可用；在 64 引脚器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。
7： 在单片机模式下，此寄存器中的位不可写且读为 0。
8： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。
9： 仅在单片机模式下的 100 引脚器件上实现。

表 5-5： 寄存器文件汇总 （PIC18F97J60 系列） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见 （页）：
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	0000 0000	60, 175
PSPCON[5]	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	61, 161
SPBRG1	EUSART1 波特率发生器寄存器的低字节								0000 0000	61, 306
RCREG1	EUSART1 接收寄存器								0000 0000	61, 313
TXREG1	EUSART1 发送寄存器								xxxx xxxx	61, 315
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	61, 306
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	61, 306
EECON2	数据存储器控制寄存器 （不是实际存在的寄存器）								---- ----	61, 96
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	---0 x00-	61, 97
IPR3	SSP2IP[5]	BCL2IP[5]	RC2IP[6]	TX2IP[6]	TMR4IP	CCP5IP	CCP4IP	CCP3IP	1111 1111	61, 132
PIR3	SSP2IF[5]	BCL2IF[5]	RC2IF[6]	TX2IF[6]	TMR4IF	CCP5IF	CCP4IF	CCP3IF	0000 0000	61, 126
PIE3	SSP2IE[5]	BCL2IE[5]	RC2IE[6]	TX2IE[6]	TMR4IE	CCP5IE	CCP4IE	CCP3IE	0000 0000	61, 129
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	1111 1-11	61, 131
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	0000 0-00	61, 125
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	0000 0-00	61, 128
IPR1	PSPIP[9]	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	1111 1111	61, 130
PIR1	PSPIF[9]	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	0000 0000	61, 124
PIE1	PSPIE[9]	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	0000 0000	61, 127
MEMCON[5,7]	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	61, 106
OSCTUNE	PPST1	PLLEN[8]	PPST0	PPRE	—	—	—	—	0000 ----	61, 41
TRISJ[6]	TRISJ7[5]	TRISJ6[5]	TRISJ5[5]	TRISJ4[5]	TRISJ3[5]	TRISJ2[5]	TRISJ1[5]	TRISJ0[5]	1111 1111	61, 159
TRISH[6]	TRISH7[6]	TRISH6[6]	TRISH5[6]	TRISH4[6]	TRISH3[6]	TRISH2[6]	TRISH1[6]	TRISH0[6]	1111 1111	61, 157
TRISG	TRISG7[5]	TRISG6[5]	TRISG5[5]	TRISG4	TRISG3[6]	TRISG2[6]	TRISG1[6]	TRISG0[6]	1111 1111	61, 155
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0[5]	1111 1111	61, 152
TRISE	TRISE7[6]	TRISE6[6]	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	1111 1111	61, 150
TRISD	TRISD7[5]	TRISD6[5]	TRISD5[5]	TRISD4[5]	TRISD3[5]	TRISD2	TRISD1	TRISD0	1111 1111	61, 147
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	61, 143
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	61, 140
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	--11 1111	61, 137
LATJ[6]	LATJ7[5]	LATJ6[5]	LATJ5[6]	LATJ4[6]	LATJ3[5]	LATJ2[5]	LATJ1[5]	LATJ0[5]	xxxx xxxx	61, 159
LATH[6]	LATH7[6]	LATH6[6]	LATH5[6]	LATH4[6]	LATH3[6]	LATH2[6]	LATH1[6]	LATH0[6]	xxxx xxxx	61, 157
LATG	LATG7[5]	LATG6[5]	LATG5[5]	LATG4	LATG3[6]	LATG2[6]	LATG1[6]	LATG0[6]	xxxx xxxx	62, 155
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0[5]	xxxx xxxx	62, 152
LATE	LATE7[6]	LATE6[6]	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx xxxx	62, 150
LATD	LATD7[5]	LATD6[5]	LATD5[5]	LATD4[5]	LATD3[5]	LATD2	LATD1	LATD0	xxxx xxxx	62, 147
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	62, 143
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	62, 140
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	00xx xxxx	62, 137
PORTJ[6]	RJ7[5]	RJ6[5]	RJ5[6]	RJ4[6]	RJ3[5]	RJ2[5]	RJ1[5]	RJ0[5]	xxxx xxxx	62, 159
PORTH[6]	RH7[6]	RH6[6]	RH5[6]	RH4[6]	RH3[6]	RH2[6]	RH1[6]	RH0[6]	0000 xxxx	62, 157
PORTG	RG7[5]	RG6[5]	RG5[5]	RG4	RG3[6]	RG2[6]	RG1[6]	RG0[6]	111x xxxx	62, 155

图注： x = 未知， u = 不变， - = 未用 （读为 0）， q = 值取决于具体条件， r = 保留位 （不能修改）。阴影单元未实现，读为 0。

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

2： PC 的 bit 21 仅在串行编程模式下可用。

3： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

4： 当 MSSP 模块工作在 I^2C™ 从动模式下时，这些位的其他名称和定义。

5： 这些位和 / 或寄存器仅在 100 引脚器件上可用；在其他器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

6： 这些位和 / 或寄存器仅在 80 引脚和 100 引脚器件上可用；在 64 引脚器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

7： 在单片机模式下，此寄存器中的位不可写且读为 0。

8： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

9： 仅在单片机模式下的 100 引脚器件上实现。

表 5-5： 寄存器文件汇总 （PIC18F97J60 系列） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见 （页）：
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0[5]	0000 0000	62, 152
PORTE	RE7[6]	RE6[6]	RE5	RE4	RE3	RE2	RE1	RE0	xxxx xxxx	62, 150
PORTD	RD7[5]	RD6[5]	RD5[5]	RD4[5]	RD3[5]	RD2	RD1	RD0	xxxx xxxx	62, 147
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	62, 143
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	62, 140
PORTA	RJPU[6]	—	RA5	RA4	RA3	RA2	RA1	RA0	0-0x 0000	62, 137
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								0000 0000	62, 306
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	62, 304
SPBRGH2	EUSART2 波特率发生器寄存器的高字节								0000 0000	62, 306
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	62, 304
ERDPTH	—	—	—	缓冲区读指针的高字节					---0 0101	62, 209
ERDPTL	缓冲区读指针的低字节								1111 1010	62, 209
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	0000 0000	62, 200
TMR4	Timer4 寄存器								0000 0000	62, 179
PR4	Timer4 周期寄存器								1111 1111	62, 179
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	62, 179
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								xxxx xxxx	62, 185
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								xxxx xxxx	62, 185
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	--00 0000	63, 181
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								xxxx xxxx	63, 185
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								xxxx xxxx	63, 185
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	--00 0000	63, 181
SPBRG2	EUSART2 波特率发生器寄存器的低字节								0000 0000	63, 306
RCREG2	EUSART2 接收寄存器								0000 0000	63, 313
TXREG2	EUSART2 发送寄存器								0000 0000	63, 315
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	63, 302
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	63, 303
ECCP3AS	ECCP3ASE	ECCP3AS2	ECCP3AS1	ECCP3AS0	PSS3AC1	PSS3AC0	PSS3BD1	PSS3BD0	0000 0000	63, 201
ECCP3DEL	P3RSEN	P3DC6	P3DC5	P3DC4	P3DC3	P3DC2	P3DC1	P3DC0	0000 0000	63, 200
ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0	0000 0000	63, 201
ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0	0000 0000	63, 200
SSP2BUF	MSSP2 接收缓冲区 / 发送寄存器								xxxx xxxx	63, 265
SSP2ADD	MSSP2 地址寄存器 （I^{2}C™ 从动模式）， MSSP2 波特率重载寄存器 （I^{2}C 主控模式）								0000 0000	63, 265
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	63, 256
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	63, 257, 267
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	63, 268
	GCEN	ACKSTAT	ADMSK5[4]	ADMSK4[4]	ADMSK3[4]	ADMSK2[4]	ADMSK1[4]	SEN		
EDATA	以太网发送 / 接收缓冲区寄存器 （EDATA<7:0>）								xxxx xxxx	63, 211
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	-000 0-00	63, 227
ECON2	AUTOINC	PKTDEC	ETHEN	—	—	—	—	—	100- ----	63, 212

图注： x = 未知， u = 不变， - = 未用 （读为 0）， q = 值取决于具体条件， r = 保留位 （不能修改）。阴影单元未实现，读为 0。

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

2： PC 的 bit 21 仅在串行编程模式下可用。

3： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

4： 当 MSSP 模块工作在 I^{2}C™ 从动模式下时，这些位的其他名称和定义。

5： 这些位和 / 或寄存器仅在 100 引脚器件上可用；在其他器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

6： 这些位和 / 或寄存器仅在 80 引脚和 100 引脚器件上可用；在 64 引脚器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

7： 在单片机模式下，此寄存器中的位不可写且读为 0。

8： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

9： 仅在单片机模式下的 100 引脚器件上实现。

 超前信息

表 5-5： 寄存器文件汇总 （PIC18F97J60 系列）（续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见（页）：
ESTAT	—	BUFER	—	LATECOL	—	RXBUSY	TXABRT	PHYRDY	-0-0 -000	63, 212
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	-000 0-00	63, 226
EDMACSH	DMA 校验和寄存器的高字节								0000 0000	63, 253
EDMACSL	DMA 校验和寄存器的低字节								0000 0000	63, 253
EDMADSTH	—	—	—	DMA 目标寄存器的高字节					---0 0000	63, 253
EDMADSTL	DMA 目标寄存器的低字节								0000 0000	63, 253
EDMANDH	—	—	—	DMA 结束寄存器的高字节					---0 0000	63, 253
EDMANDL	DMA 结束寄存器的低字节								0000 0000	63, 253
EDMASTH	—	—	—	DMA 开始寄存器的高字节					---0 0000	63, 253
EDMASTL	DMA 开始寄存器的低字节								0000 0000	63, 253
ERXWRPTH	—	—	—	接收缓冲区写指针的高字节					---0 0000	63, 241
ERXWRPTL	接收缓冲区写指针的低字节								0000 0000	63, 241
ERXRDPTH	—	—	—	接收缓冲区读指针的高字节					---0 0101	63, 241
ERXRDPTL	接收缓冲区读指针的低字节								1111 1010	63, 241
ERXNDH	—	—	—	接收结束寄存器的高字节					---1 1111	63, 241
ERXNDL	接收结束寄存器的低字节								1111 1111	63, 241
ERXSTH	—	—	—	接收开始寄存器的高字节					---0 0101	63, 241
ERXSTL	接收开始寄存器的低字节								1111 1010	63, 241
ETXNDH	—	—	—	发送结束寄存器的高字节					---0 0000	64, 241
ETXNDL	发送结束寄存器的低字节								0000 0000	64, 241
ETXSTH	—	—	—	发送开始寄存器的高字节					---0 0000	64, 241
ETXSTL	发送开始寄存器的低字节								0000 0000	64, 241
EWRPTH	—	—	—	缓冲区写指针的高字节					---0 0000	64, 209
EWRPTL	缓冲区写指针的低字节								0000 0000	64, 209
EPKTCNT	以太网数据包计数寄存器								0000 0000	64, 241
ERXFCON	UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN	1010 0001	64, 246
EPMOH	—	—	—	模式匹配偏移寄存器的高字节					---0 0000	64, 249
EPMOL	模式匹配偏移寄存器的低字节								0000 0000	64, 249
EPMCSH	模式匹配校验和寄存器的高字节								0000 0000	64, 249
EPMCSL	模式匹配校验和寄存器的低字节								0000 0000	64, 249
EPMM7	模式匹配屏蔽寄存器的字节 7								0000 0000	64, 249
EPMM6	模式匹配屏蔽寄存器的字节 6								0000 0000	64, 249
EPMM5	模式匹配屏蔽寄存器的字节 5								0000 0000	64, 249
EPMM4	模式匹配屏蔽寄存器的字节 4								0000 0000	64, 249
EPMM3	模式匹配屏蔽寄存器的字节 3								0000 0000	64, 249
EPMM2	模式匹配屏蔽寄存器的字节 2								0000 0000	64, 249
EPMM1	模式匹配屏蔽寄存器的字节 1								0000 0000	64, 249
EPMM0	模式匹配屏蔽寄存器的字节 0								0000 0000	64, 249
EHT7	哈希表寄存器的字节 7								0000 0000	64, 249
EHT6	哈希表寄存器的字节 6								0000 0000	64, 245
EHT5	哈希表寄存器的字节 5								0000 0000	64, 245

图注： x = 未知， u = 不变， - = 未用 （读为 0）， q = 值取决于具体条件， r = 保留位 （不能修改）。阴影单元未实现，读为 0。

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

2： PC 的 bit 21 仅在串行编程模式下可用。

3： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

4： 当 MSSP 模块工作在 I^2C™ 从动模式下时，这些位的其他名称和定义。

5： 这些位和 / 或寄存器仅在 100 引脚器件上可用；在其他器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

6： 这些位和 / 或寄存器仅在 80 引脚和 100 引脚器件上可用；在 64 引脚器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

7： 在单片机模式下，此寄存器中的位不可写且读为 0。

8： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

9： 仅在单片机模式下的 100 引脚器件上实现。

表 5-5： 寄存器文件汇总 （**PIC18F97J60** 系列） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见（页）：
EHT4	哈希表寄存器的字节 4								0000 0000	64, 245
EHT3	哈希表寄存器的字节 3								0000 0000	64, 245
EHT2	哈希表寄存器的字节 2								0000 0000	64, 245
EHT1	哈希表寄存器的字节 1								0000 0000	64, 245
EHT0	哈希表寄存器的字节 0								0000 0000	64, 245
MIRDH	MII 读数据寄存器的高字节								0000 0000	64, 217
MIRDL	MII 读数据寄存器的低字节								0000 0000	64, 217
MIWRH	MII 写数据寄存器的高字节								0000 0000	64, 217
MIWRL	MII 写数据寄存器的低字节								0000 0000	64, 217
MIREGADR	—	—	—	MII 地址寄存器					---0 0000	64, 217
MICMD	—	—	—	—	—	—	MIISCAN	MIIRD	---- --00	64, 216
MICON	RSTMII	—	—	—	—	—	—	—	0--- ----	64, 215
MAMXFLH	最大帧长寄存器的高字节								0000 0110	64, 241
MAMXFLL	最大帧长寄存器的低字节								0000 0000	64, 241
MACLCON2	—	—	冲突窗口寄存器						--11 0111	64, 241
MACLCON1	—	—	—	—	重发最大数寄存器				---- 1111	64, 241
MAIPGH	—	非背对背包间间隔寄存器的高字节							-000 0000	65, 241
MAIPGL	—	非背对背包间间隔寄存器的低字节							-000 0000	65, 241
MABBIPG	—	背对背包间间隔发送寄存器							-000 0000	65, 232
MACON4	—	DEFER	BPEN	NOBKOFF	—	—	r	r	-000 --00	65, 215
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDRLEN	HFRMEN	FRMLNEN	FULDPX	0000 0000	65, 214
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	---0 0000	65, 213
EPAUSH	暂停定时器值寄存器的高字节								0001 0000	65, 244
EPAUSL	暂停定时器值寄存器的低字节								0000 0000	65, 244
EFLOCON	—	—	—	—	—	FULDPXS	FCEN1	FCEN0	---- -000	65, 244
MISTAT	—	—	—	—	r	NVALID	SCAN	BUSY	---- 0000	65, 216
MAADR2	MAC 地址寄存器的字节 2 （MAADR<39:32>）， OUI 字节 2								0000 0000	65, 231
MAADR1	MAC 地址寄存器的字节 1 （MAADR<47:40>）， OUI 字节 1								0000 0000	65, 231
MAADR4	MAC 地址寄存器的字节 4 （MAADR<23:16>）								0000 0000	65, 231
MAADR3	MAC 地址寄存器的字节 3 （MAADR<31:24>）， OUI 字节 3								0000 0000	65, 231
MAADR6	MAC 地址寄存器的字节 6 （MAADR<7:0>）								0000 0000	65, 231
MAADR5	MAC 地址寄存器的字节 5 （MAADR<15:8>）								0000 0000	65, 231

图注： x = 未知， u = 不变， - = 未用 （读为 0）， q = 值取决于具体条件， r = 保留位 （不能修改）。阴影单元未实现，读为 0。

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

2： PC 的 bit 21 仅在串行编程模式下可用。

3： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

4： 当 MSSP 模块工作在 I^{2}C™ 从动模式下时，这些位的其他名称和定义。

5： 这些位和 / 或寄存器仅在 100 引脚器件上可用；在其他器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

6： 这些位和 / 或寄存器仅在 80 引脚和 100 引脚器件上可用；在 64 引脚器件上，它们未实现并读为 0。所示为 100 引脚器件的复位值。

7： 在单片机模式下，此寄存器中的位不可写且读为 0。

8： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

9： 仅在单片机模式下的 100 引脚器件上实现。

 超前信息

5.3.6 STATUS 寄存器

如寄存器 5-3 所示，STATUS 寄存器包含 ALU 的算术运算状态。STATUS 寄存器与任何其他寄存器一样，可以作为任何指令的操作数。如果一条影响 Z、DC、C、OV 或 N 位的指令以 STATUS 寄存器作为目标寄存器，则会禁止对这 5 位进行写操作。

这些位根据器件逻辑来置 1 或清零。因此，当执行一条把 STATUS 寄存器作为目标寄存器的指令后，运行结果可能会与预想的不同。例如， CLRF STATUS 会将 Z 位置 1，而保留其余位不变。然后读 STATUS 寄存器将得到 “000u u1uu” 。因此，建议仅使用 BCF、 BSF、SWAPF、MOVFF 和 MOVWF 指令来改变 STATUS 寄存器，因为这些指令不会影响STATUS寄存器中的Z、C、DC、OV 或 N 位。

关于其他不会影响状态位的指令，请参见表 25-2和表 25-3 中的指令集汇总。

> 注： 在减法运算中，C 和 DC 位分别作为借位位和半借位位。

寄存器 5-3： STATUS 寄存器

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC(1)	C(2)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-5 未用：读为 0

bit 4 **N**：负标志位
此位用于有符号的算术运算 （2 进制补码）。它可以表示结果是否为负 （ALU MSb = 1）。
1 = 结果为负
0 = 结果为正

bit 3 **OV**：溢出标志位
此位用于有符号的算术运算 （2 进制补码）。表明运算结果溢出了 7 位二进制数的范围，溢出导致符号位 （bit 7）发生改变。
1 = 有符号算术运算中发生溢出 （本次运算）
0 = 未发生溢出

bit 2 **Z**：全零标志位
1 = 算术运算或逻辑运算的结果为零
0 = 算术运算或逻辑运算的结果不为零

bit 1 **DC**：半进位 / $\overline{借位}$位 (1)
用于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
1 = 结果的第 4 个低位发生了进位
0 = 结果的第 4 个低位未发生进位

bit 0 **C**：进位 / $\overline{借位}$位 (2)
用于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
1 = 结果中最高位发生了进位
0 = 结果中最高位未发生进位

注 1： 对于$\overline{借位}$，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来执行的。对于移位指令 （RRF 和 RLF），此位来自源寄存器的 bit 4 或 bit 3。

2： 对于$\overline{借位}$，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来执行的。对于移位指令 （RRF 和 RLF），此位来自源寄存器的最高位或最低位。

5.4 数据寻址模式

> **注：** 当使能 PIC18 扩展指令集时， PIC18 内核指令集中某些指令的执行方式会发生改变。更多信息，请参见**第 5.6 节“数据存储器和扩展指令集”**。

程序存储器只能用一种方式寻址 （通过程序计数器），而数据存储器空间可用多种方式寻址。大部分指令的寻址模式都是固定的。其他指令可能使用最多三种模式，根据它们所使用的操作数和是否使能了扩展指令集而定。

这些寻址模式为：

- 固有寻址
- 立即数寻址
- 直接寻址
- 间接寻址

当使能了扩展指令集时 （XINST 配置位 = 1），还可使用另外一种寻址模式，即立即数变址寻址模式。**第 5.6.1 节“使用立即数偏移量进行变址寻址”**将更详细讨论它的操作。

5.4.1 固有和立即数寻址

很多 PIC18 控制指令根本不需要任何参数。执行这些指令要么对整个器件造成影响，要么仅针对一个寄存器进行操作。此寻址模式就是固有寻址。例如指令 SLEEP、RESET 和 DAW。

其他指令的工作方式与此类似，但需要操作码中有其他的参数。由于需要一些立即数作为参数，这种寻址模式被称为立即数寻址。例如 ADDLW 和 MOVLW，它们分别向 W 寄存器添加或移入立即数值。其他立即数寻址指令，例如 CALL 和 GOTO，它们包括 20 位的程序存储器地址。

5.4.2 直接寻址

直接寻址方式在操作码中指定操作的全部或部分源地址和 / 或目标地址。此选项由指令附带的参数指定。

在 PIC18 内核指令集中，针对位和针对字节的指令默认情况下使用直接寻址模式。所有这些指令都包含某个 8 位的直接地址作为它们的最低有效字节。此地址指定数据 RAM 的某个存储区中寄存器的地址（**第 5.3.3 节“通用寄存器文件”**）或快速操作存储区 （**第 5.3.2 节 “快速操作存储区”**）中作为指令数据源的单元地址。

快速操作 RAM 位 “a”决定地址的解析方式。当 “a”为 1 时，BSR（**第 5.3.1 节“存储区选择寄存器”**）的内容将和指令中的直接地址一起用于确定寄存器的完整 12 位地址。当 “a”为 0 时，此直接地址将被解析为快速操作存储区中的一个寄存器。使用快速操作 RAM 的寻址模式有时也被称为直接强制寻址模式。

有几个指令，例如 MOVFF，在操作码中包含完整的 12 位地址（源地址或目标地址）。在这些情况下，BSR 被完全忽略。

操作目标地址由目标位 “d”确定。当 “d”为 1 时，结果被存回源寄存器并覆盖原来的内容。当 “d”为 0 时，结果被存储在 W 寄存器中。没有 “d”参数的指令的目标地址是隐含的，它们是正在操作的目标寄存器或 W 寄存器。

5.4.3 间接寻址

间接寻址模式允许用户访问数据存储器中的单元而无需在指令中给出一个固定的地址。这种寻址方式是通过使用文件选择寄存器 （File Select Register， FSR）作为指向被读写单元的指针实现的。由于FSR本身作为特殊功能寄存器位于 RAM 中，因此也可在程序中直接控制它们。这使得FSR对于在数据存储器中实现诸如表和数组等数据结构时非常有用。

也可以使用间接文件操作数 （Indirect File Operand，INDF）进行间接寻址。这种操作允许自动递增、递减或偏移指针，从而自动控制指针的值。它通过使用循环提高代码执行效率，如例 5-5 所示的清零整个 RAM 存储区的操作。它还允许用户在数据存储器中执行变址寻址和其他针对程序存储器堆栈指针的操作。

例 5-5： 使用间接寻址清零 RAM （BANK 1）

```
        LFSR  FSR0, 100h  ;
NEXT    CLRF  POSTINC0    ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS FSR0H, 1    ; All done with
                          ; Bank1?
        BRA   NEXT        ; NO, clear next
CONTINUE                  ; YES, continue
```

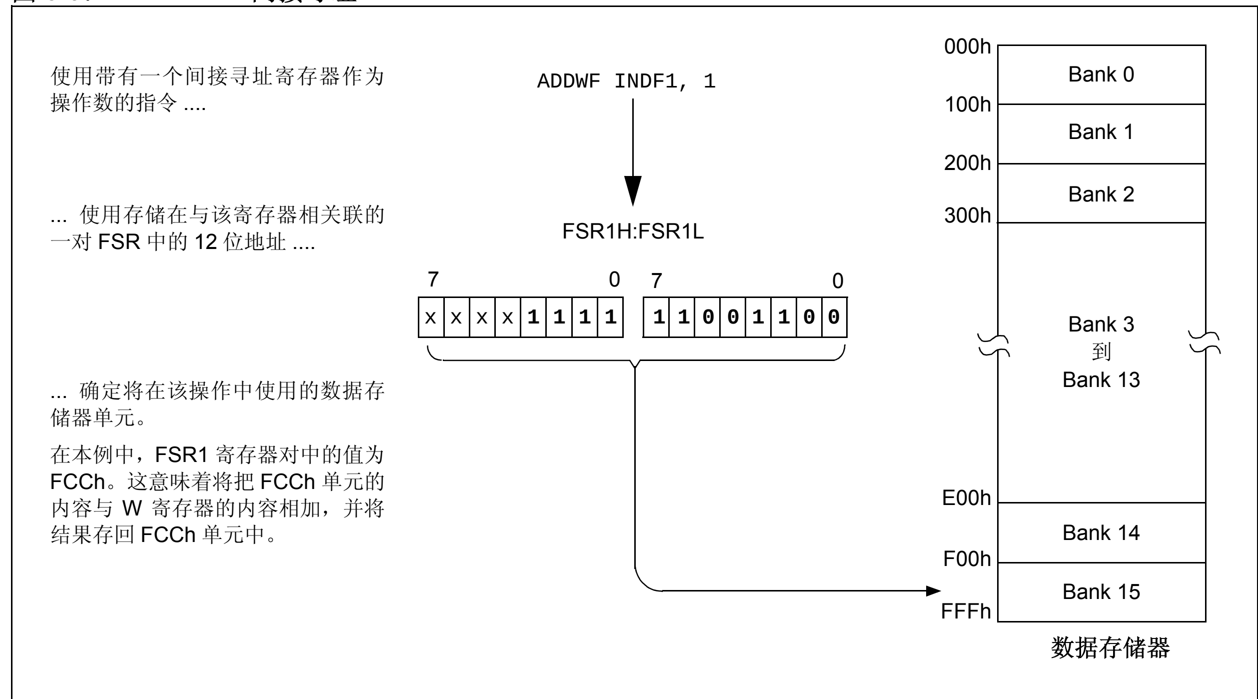

5.4.3.1 FSR 寄存器和 INDF 操作数

间接寻址的核心是三组寄存器：FSR0、FSR1和FSR2。每组寄存器都含有一对 8 位寄存器：FSRnH 和 FSRnL。FSRnH 寄存器的高 4 位未使用，因此每对 FSR 只保存一个 12 位值，从而可以线性寻址数据存储器的整个空间。因此， FSR 寄存器对被用作数据存储器的地址指针。

间接寻址是通过一组间接文件操作数（INDF0到INDF2）完成的。这些操作数可被看作 “虚拟”寄存器：它们被映射到 SFR 空间而不是通过物理方式实现的。对特定的 INDF 寄存器执行读或写操作实际上访问的是与之对应的一对 FSR 寄存器。例如，读 INDF1 就是读 FSR1H:FSR1L 指向单元中的数据。使用 INDF 寄存器作为操作数的指令实际上使用的是相应的 FSR 的内容，该内容为指向目标地址的指针。INDF 操作数只是使用该指针的一种简便方法。

由于间接寻址使用完整的 12 位地址，因此没有必要进行数据 RAM 分区。所以 BSR 的当前内容和快速操作 RAM 位对于确定目标地址没有影响。

图 5-9： 间接寻址



5.4.3.2 FSR 寄存器和 POSTINC、 POSTDEC、 PREINC 以及 PLUSW

除了 INDF 操作数之外，每对 FSR 寄存器还有 4 个额外的间接操作数。和 INDF 一样，它们也都是不能直接读写的 “虚拟” 寄存器。访问这些寄存器实际上访问的是与之相关的一对 FSR 寄存器，并对其所存储的数值进行特定的操作。这些寄存器是：

- POSTDEC：访问 FSR 的值，然后将它自动减 1
- POSTINC：访问 FSR 的值，然后将它自动加 1
- PREINC：将 FSR 的值加 1，然后在操作中使用该值
- PLUSW：将 W 寄存器中有符号的值 （-128 到 127）与 FSR 寄存器中的值相加，并在操作中使用得到的新值

在本文中使用 FSR 寄存器中的值 （不会更改此值）访问 INDF 寄存器。同样，访问 PLUSW 寄存器是将 W 寄存器中的值作为 FSR 的偏移量。该操作不会改变这两个寄存器中的值，而访问其他虚拟寄存器均会更改 FSR 寄存器的值。

使用 POSTDEC、 POSTINC 和 PREINC 对 FSR 进行操作会影响整对寄存器：即， FSRnL 寄存器从 FFh 到 00h 溢出并向 FSRnH 寄存器进位。但这些操作的结果不会更改状态寄存器中的标志位（如 Z、N 和 OV 等）。

PLUSW 寄存器可用于在数据存储器空间实现变址寻址。通过控制 W 寄存器中的值，用户可以访问相对当前指针地址有固定偏移量的地址单元。在某些应用中，该功能可被用于在数据存储器内部实现某些强大的程序控制结构，如软件堆栈。

5.4.3.3 通过 FSR 对其他 FSR 进行操作

在某些特殊情况下，间接寻址操作以其他 FSR 或虚拟寄存器作为目标。例如，使用 FSR 指向一个虚拟寄存器会导致操作不成功。假设如下特殊情况 ：FSR0H:FSR0L 对保存的是 INDF1 的地址 FE7h。尝试使用 INDF0 作为操作数读取 INDF1 的值，将返回 00h。尝试使用 INDF0 作为操作数写入 INDF1，将会导致执行一条 NOP 指令。

另一方面，使用虚拟寄存器对一对 FSR 寄存器进行写操作可能会产生与预期不同的结果。在这些情况下，会将值写入一对 FSR 寄存器，但 FSR 不会递增或递减。因此，写入 INDF2 或 POSTDEC2 时会把同样的值写入 FSR2H:FSR2L 对。

由于 FSR 是映射到 SFR 空间中的物理寄存器，所以可以通过直接寻址来控制它们。用户在使用这些寄存器时应该特别小心，尤其是在代码使用间接寻址的方式时。

同样，通常允许通过间接寻址对所有其他 SFR 进行操作。用户在进行此类操作时应该特别小心，以免不小心更改设置从而影响器件操作。

5.5 程序存储器和扩展指令集

程序存储器的操作不受扩展指令集的影响。

使能扩展指令集会将 5 条额外的双字命令添加到现有的 PIC18 指令集中：即 ADDFSR、CALLW、MOVSF、MOVSS 和 SUBFSR。这些指令如**第 5.2.4 节 “双字指令”**中所述执行。

5.6 数据存储器和扩展指令集

使能 PIC18 扩展指令集（XINST 配置位 = 1）显著改变了数据存储器及其寻址的某些方面。特别是许多 PIC18 内核指令使用快速操作存储区的方式有所不同。这是由于扩展指令集引入了对数据存储器空间的新的寻址模式。该模式还会更改使用 FSR2 及其相关操作数进行间接寻址的方式。

同样需要了解哪些部分保持不变。数据存储器空间的大小及其线性寻址方式都不会改变。 SFR 映射也保持不变。PIC18 内核指令也仍然以直接和间接寻址模式进行操作；固有和立即数指令操作照旧。 FSR0 和 FSR1 的间接寻址方式也保持不变。

5.6.1 使用立即数偏移量进行变址寻址

使能 PIC18 扩展指令集将更改使用 FSR2 寄存器对及其相关文件操作数进行间接寻址的方式。在适当的条件下，使用快速操作存储区的指令（即绝大多数针对位和针对字节的指令）可以利用指令中的偏移量来执行变址寻址。这种特定的寻址模式被称为使用立即数偏移量的变址寻址或立即数变址寻址模式。

使用扩展指令集时，这种寻址模式有如下要求：

- 强制使用快速操作存储区 （a = 0）；且
- 文件地址参数要小于或等于 5Fh。

在这些条件下，指令的文件地址不会被解析为地址的低字节 （在直接寻址中和 BSR 一起使用），或快速操作存储区中的 8 位地址，而是被解析为由 FSR2 指定的地址指针的偏移量。将该偏移量与 FSR2 的内容相加以获取操作的目标地址。

5.6.2 受立即数变址寻址模式影响的指令

任何采用直接寻址模式的 PIC18 内核指令均会受到立即数变址寻址模式的潜在影响，包括所有针对字节和针对位的指令，或标准 PIC18 指令集中几乎一半的指令。只有使用固有或立即数寻址模式的指令不受影响。

此外，如果针对字节和针对位的指令使用快速操作存储区 （快速操作 RAM 位为 1）或包含 60h 以上的文件地址，它们也不受影响。符合这些条件的指令会像以前一样执行。图 5-10 显示了当使能扩展指令集时，各种寻址模式之间的对比。

那些想要在立即数变址寻址模式中使用针对位或针对字节的指令的用户，应该注意此模式下汇编语法的改变。在第 **25.2.1 节 “扩展指令的语法”**中对此进行了更详细的说明。

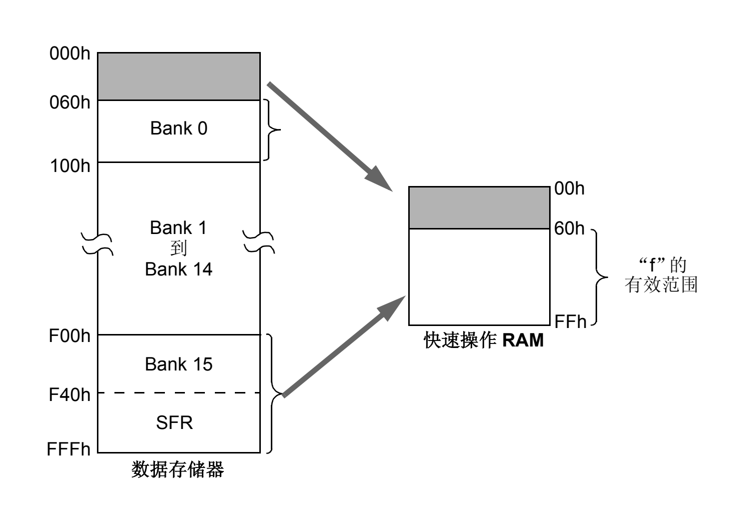
图 5-10： 针对位和针对字节的指令的寻址方式对比 （使能了扩展指令集）

示例指令：`ADDWF f, d, a`（操作码：`0010 01da ffff ffff`）

当 a = 0 且 f ≥ 60h 时：

此指令以直接强制模式执行。“f”被解析为快速操作 RAM 中 060h 到 FFFh 之间的单元地址，该地址也是数据存储器的 F60h 到 FFFh （Bank 15）。

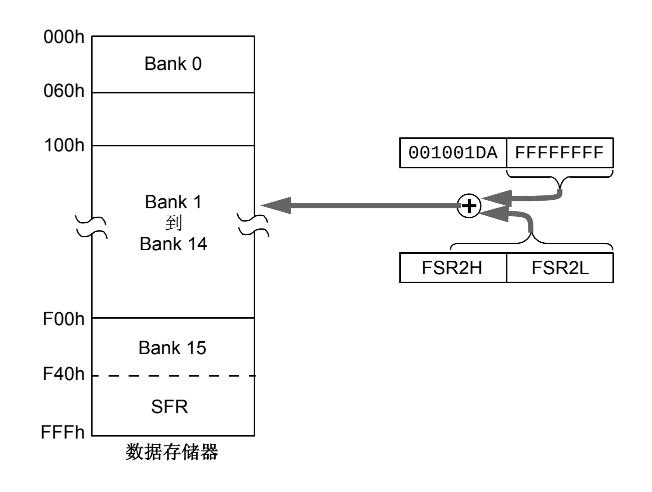
不可用此模式寻址地址低于 060h 的单元。



当 a = 0 且 f ≤ 5Fh 时：

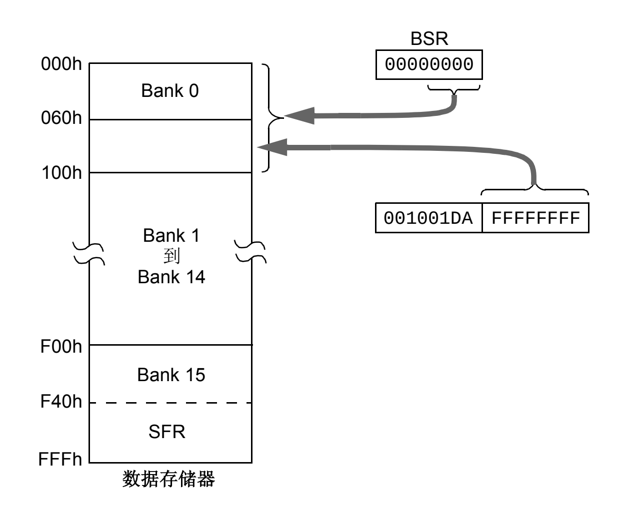
此指令以立即数变址寻址模式执行。“f”被解析为 FSR2 中地址值的偏移量。将这两个值相加可以得到指令的目标寄存器的地址。此地址可以在数据存储器空间的任何地方。

注意在此模式中，正确的语法如下：
`ADDWF [k], d`
其中 “k” 就是 “f”。



当 a = 1（f 可为任何值）时：

指令以直接寻址模式 （也称为直接长地址寻址模式）执行。“f” 被解析为数据存储器空间的 16 个存储区中的一个单元地址。存储区由存储区选择寄存器 （BSR）指定。此地址可以在数据存储器空间的任何地方。



 超前信息

5.6.3 在立即数变址模式中映射快速操作存储区

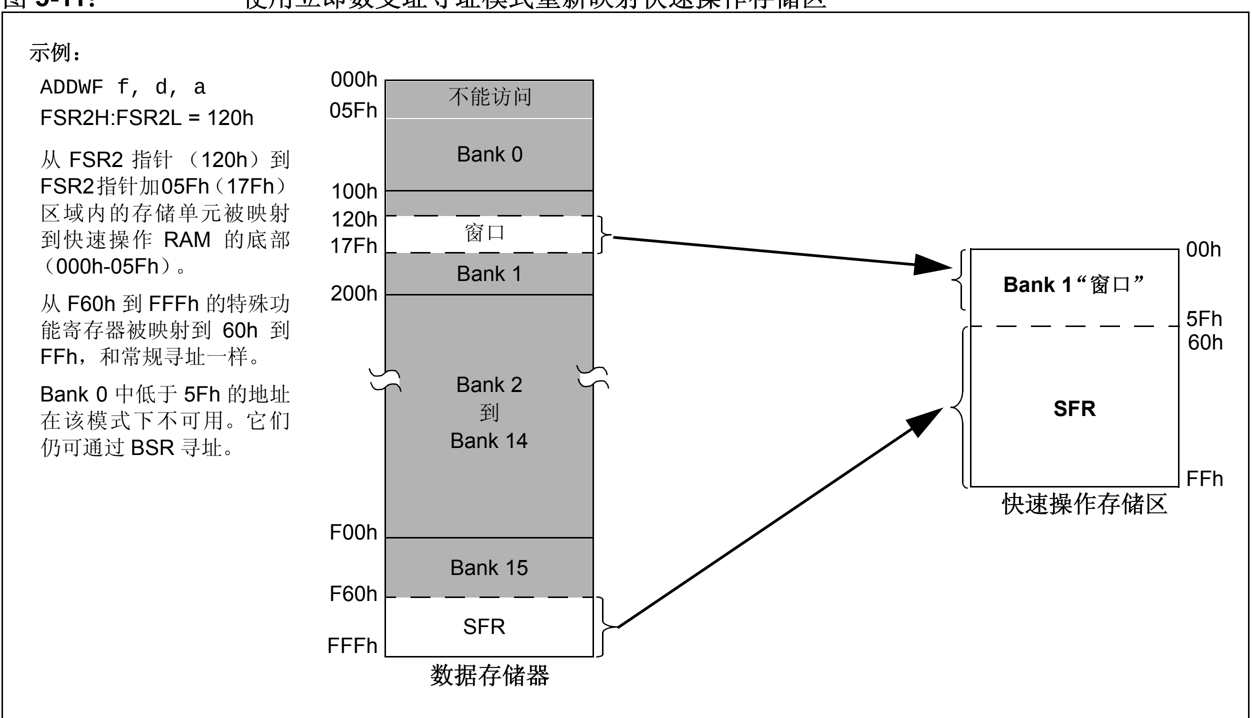
使用立即数变址寻址模式能有效改变快速操作 RAM 低地址单元（00h 到 5Fh）的映射方式。此模式映射 Bank 0 的内容和由用户定义的、可位于数据存储器空间中任何地方的 “窗口” 内容，而不仅仅映射 Bank 0 底部的内容。FSR2 的值定义映射到窗口的地址的下边界，而上边界则由 FSR2 加 95 （5Fh）决定。地址为 5Fh 以上的快速操作 RAM 的映射方法如前所述 （见第 **5.3.2 节 “快速操作存储区”**）。图 5-11 显示了在此寻址模式下重新映射的快速操作存储区示例。

快速操作存储区的重新映射*仅*适用于立即数变址寻址模式。使用 BSR （快速操作 RAM 位为 `1`）的操作和以前一样继续使用直接寻址模式。任何明确使用间接文件操作数 （包括 FSR2）的间接或变址操作都将像标准间接寻址一样操作。任何使用快速操作存储区的指令（包含大于 05Fh 的寄存器地址）都将使用直接寻址和常规的快速操作存储区映射。

5.6.4 立即数变址模式中的 BSR

尽管使能扩展指令集时会重新映射快速操作存储区，但 BSR 的操作不变。使用 BSR 选择数据存储区的直接寻址操作方式和以前描述的相同。

图 5-11： 使用立即数变址寻址模式重新映射快速操作存储区



注：

6.0 闪存程序存储器

正常工作状态下，闪存程序存储器在整个 VDD 范围内都是可读写可擦除的。

读程序存储器时，每次读取一个字节。写程序存储器时，每次写入一个 64 字节的块。擦除程序存储器时，每次擦除一个 1024 字节的块。用户代码不能执行批量擦除操作。

在擦写程序存储器时，系统会停止取指令直到操作完成。擦写期间不能访问该程序存储器，因此也就无法执行代码。由内部编程定时器来终止程序存储器的擦写操作。

写入程序存储器的值不一定非要是有效指令。执行存储无效指令的程序存储器单元会导致执行 NOP。

6.1 表读与表写

为了读写程序存储器，有两个操作指令可供处理器在程序存储器空间和数据 RAM 之间移动字节：

- 表读 （TBLRD）
- 表写 （TBLWT）

程序存储器空间为 16 位宽，而数据 RAM 空间为 8 位宽。表读和表写操作通过一个 8 位寄存器 （TABLAT）在这两个存储器空间之间传送数据。

表读操作从程序存储器获取数据并将其放入数据 RAM 空间。图 6-1 显示了程序存储器和数据 RAM 之间的一次表读操作。

表写操作将数据存储器中的数据存入程序存储器的保持寄存器。**第 6.5 节 "写闪存程序存储器"**详细介绍了将保持寄存器内容写入程序存储器的过程。图 6-2 显示了程序存储器和数据 RAM 之间的一次表写操作。

表操作以字节为单位。包含数据而非程序指令的表块不需要按字对齐。因此，表块可以在任何字节地址开始和结束。如果使用表写操作向程序存储器写入可执行代码，程序指令必须按字对齐。

图 6-1： 表读操作

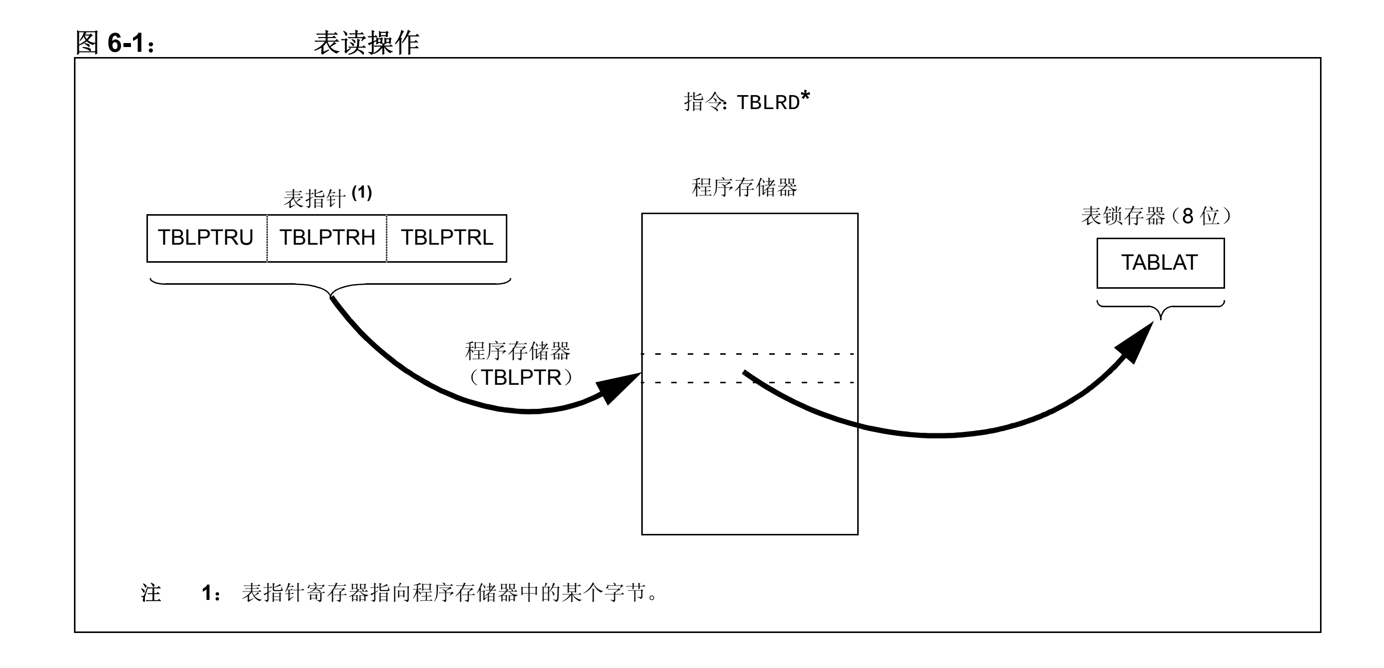


图 **6-2**： 表写操作

注 1： 表指针实际指向 64 个保持寄存器中的某一个，其地址由 TBLPTRL<5:0> 决定。第 **6.5** 节 **"写闪存程序存储器"**中将讨论向程序存储器阵列写入数据的物理过程。

6.2 控制寄存器

TBLRD 和 TBLWT 指令要用到几个控制寄存器。包括：

- EECON1 寄存器
- EECON2 寄存器
- TABLAT 寄存器
- TBLPTR 寄存器

6.2.1 EECON1 和 EECON2 寄存器

EECON1 寄存器（寄存器 6-1）是存储器访问的控制寄存器。EECON2 寄存器不是实际存在的寄存器，专用于存储器的擦写操作。读 EECON2 将得到全 0。

当 FREE 位置 1 时，允许对程序存储器进行擦除操作，擦除操作由下一条 WR 命令启动。当 FREE 清零时，则仅使能写操作。

当 WREN 位置 1 时，允许进行写操作。上电时，WREN 位被清零。WRERR 位在 WR 位置 1 时由硬件置 1，在内部编程定时器超时、写操作结束时被清零。

注：	在正常操作期间，WRERR 读为 1。这表明写操作被复位提早终止或进行了不合法的写操作。

WR 控制位用于启动写操作。用软件只能将该位置 1 而无法清零。在写操作完成后，由硬件将其清零。

寄存器 6-1： EECON1：EEPROM 控制寄存器 1

U-0	U-0	U-0	R/W-0	R/W-x	R/W-0	R/S-0	U-0
—	—	—	FREE	WRERR	WREN	WR	—
bit 7							bit 0

图注：	S = 可设置位		
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-5 未用：读为 0

bit 4 **FREE**：闪存行擦除使能位
1 = 在下一条 WR 命令时擦除 TBLPTR 指定的程序存储器行 （擦除操作完成后清零）
0 = 仅执行写操作

bit 3 **WRERR**：闪存程序存储器错误标志位
1 = 写操作提早终止 （由于正常操作中自定时编程期间的任何复位，或不合法的写操作）
0 = 写操作完成

bit 2 **WREN**：闪存程序存储器写使能位
1 = 允许对闪存程序存储器的写周期
0 = 禁止对闪存程序存储器的写周期

bit 1 **WR**：写控制位
1 = 启动程序存储器的擦写周期
（操作是自定时的，一旦写操作完成，该位即由硬件清零。
用软件只能将 WR 位置 1，但不能清零。）
0 = 写周期完成

bit 0 未用：读为 0

6.2.2 表锁存寄存器 （TABLAT）

表锁存器（TABLAT）是映射到 SFR 空间的一个 8 位寄存器。表锁存器用于在程序存储器和数据 RAM 之间传输数据时保存 8 位数据。

6.2.3 表指针寄存器 （TBLPTR）

表指针 （TBLPTR）寄存器在程序存储器中以字节为单位进行寻址。 TBLPTR 由 3 个 SFR 寄存器组成：表指针最高字节、表指针高字节和表指针低字节（TBLPTRU:TBLPTRH:TBLPTRL）。这 3 个寄存器合起来组成一个 22 位宽的指针。其中低 21 位允许器件寻址高达 2MB 程序存储器空间。第 22 位则允许访问器件 ID 和配置位。

TBLRD 和 TBLWT 指令要使用表指针寄存器 TBLPTR。这些指令可以基于表操作以 4 种方法更新 TBLPTR。表 6-1 列出了这些操作。这些表操作只会影响 TBLPTR 的低 21 位。

6.2.4 表指针范围

TBLPTR 用于读、写和擦除闪存程序存储器。

当执行 TBLRD 时，TBLPTR 的所有 22 位决定将程序存储器的哪个字节读入 TABLAT。

当执行 TBLWT 时，表指针寄存器的低 6 位（TBLPTR<5:0>）决定要写入程序存储器的哪个保持寄存器 （共有 64 个）。当程序存储器的定时写入 （通过 WR 位）开始时，TBLPTR 的高 15 位（TBLPTR<20:6>）将决定要写入哪个程序存储器块 （每块 64 字节）。更多详细信息，请参见**第 6.5 节 "写闪存程序存储器"**。

当执行擦除程序存储器时，表指针寄存器的高 11 位（TBLPTR<20:10>）指向将要擦除的 1024 字节块。低有效位 （TBLPTR<9:0>）被忽略。

图 6-3 说明了基于闪存程序存储器操作的 TBLPTR 相关范围。

表 6-1： 执行 TBLRD 和 TBLWT 指令的表指针操作

示例	表指针操作
TBLRD* TBLWT*	不修改 TBLPTR
TBLRD*+ TBLWT*+	TBLPTR 在读 / 写后递增
TBLRD*- TBLWT*-	TBLPTR 在读 / 写后递减
TBLRD+* TBLWT+*	TBLPTR 在读 / 写前递增

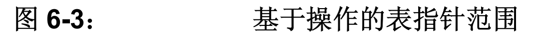
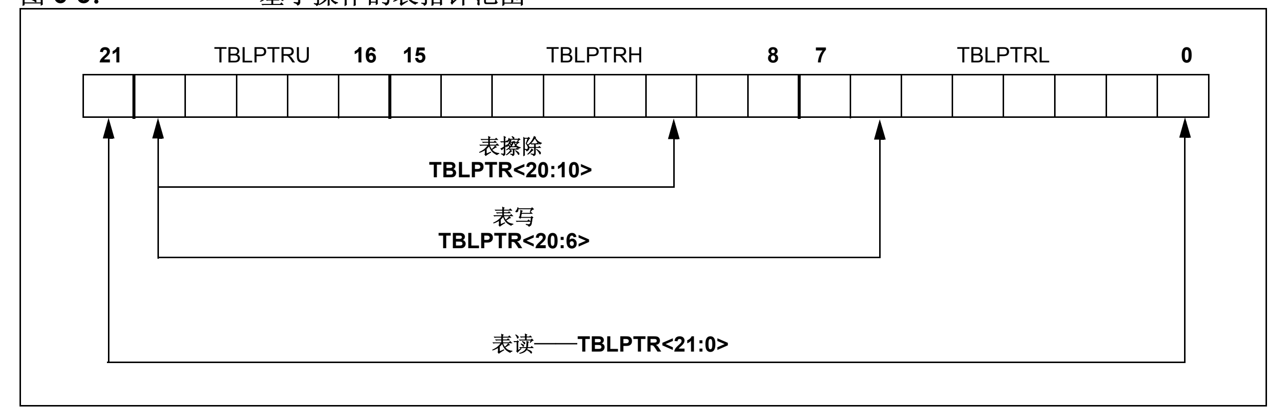


图 6-3： 基于操作的表指针范围



 超前信息

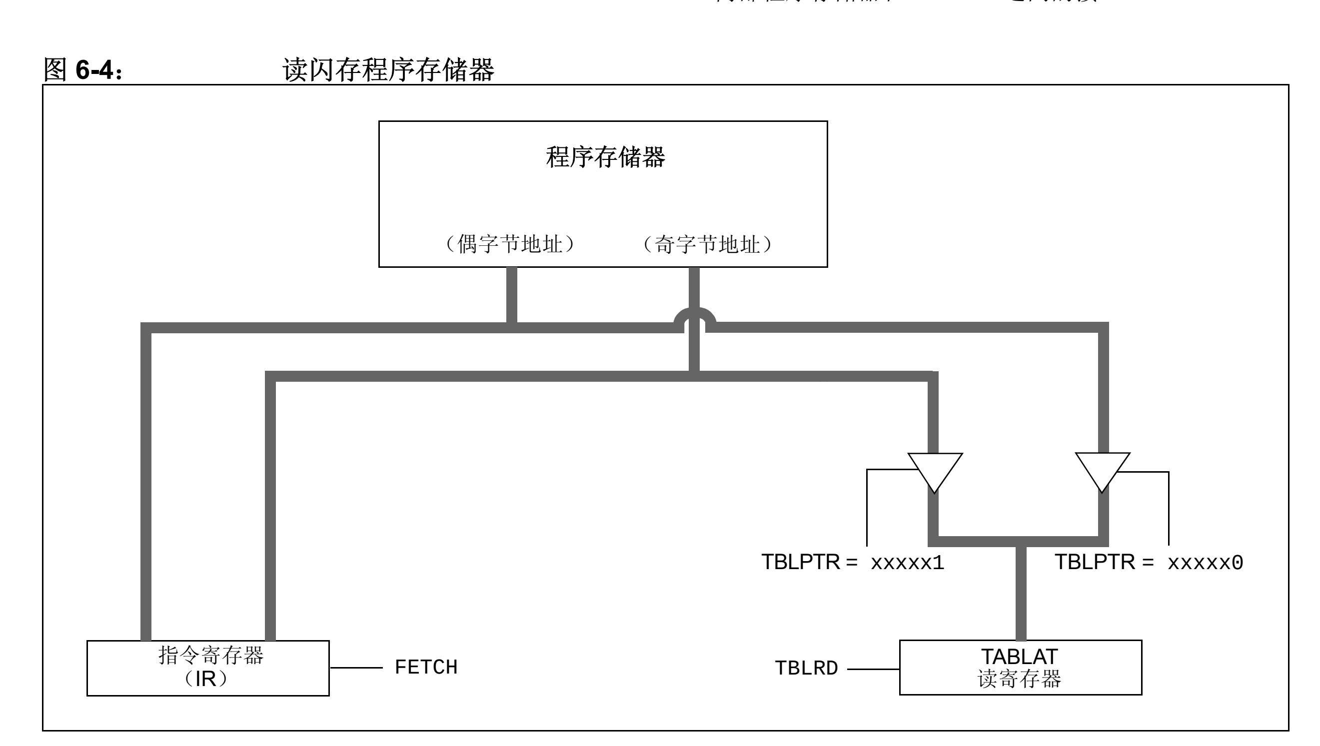
6.3 读闪存程序存储器

TBLRD 指令用于从程序存储器获取数据并放入数据 RAM。表读操作每次从程序存储器读取一个字节。

TBLPTR 指向程序存储器空间的某个字节地址。执行 TBLRD 指令将把指向的字节装入 TABLAT。此外，还可以自动修改 TBLPTR 以进行下一次表读操作。

内部程序存储器通常以字为单位进行组织。由地址的最低有效位来选择字的高字节或者低字节。图 6-4 显示了内部程序存储器和 TABLAT 之间的接口。

图 6-4： 读闪存程序存储器



例 6-1： 读闪存程序存储器的一个字

```
          MOVLW     CODE_ADDR_UPPER        ; Load TBLPTR with the base
          MOVWF     TBLPTRU                ; address of the word
          MOVLW     CODE_ADDR_HIGH
          MOVWF     TBLPTRH
          MOVLW     CODE_ADDR_LOW
          MOVWF     TBLPTRL
READ_WORD
          TBLRD*+                          ; read into TABLAT and increment
          MOVF      TABLAT, W              ; get data
          MOVWF     WORD_EVEN
          TBLRD*+                          ; read into TABLAT and increment
          MOVFW     TABLAT, W              ; get data
          MOVF      WORD_ODD
```

6.4 擦除闪存程序存储器

最小擦除块大小为 1024 字节。只有通过使用外部编程器，或通过 ICSP 控制，才能够批量擦除更大的程序存储器块。闪存阵列不支持字擦除。

当单片机本身开始一个擦除过程时，会擦除一个1024字节的程序存储器块。 11 个最高有效位 TBLPTR<20:10> 指向要擦除的块。 TBLPTR<9:0> 被忽略。

擦除操作由 EECON1 寄存器控制。WREN 位必须被置 1 以使能写操作。 FREE 位被置 1 以选择擦除操作。

为了安全起见，必须使用 EECON2 的写启动序列。

擦除内部闪存必须执行长写操作。在长写周期中，指令停止执行。由内部编程定时器终止长写操作。

6.4.1 闪存程序存储器擦除序列

擦除内部程序存储器块的步骤如下：

1. 将要擦除的行地址装入表指针寄存器。
2. 设置 EECON1 寄存器来执行擦除操作：
   - 将 WREN 位置 1 以使能写操作；
   - 将 FREE 位置 1 以使能擦除操作。
3. 禁止中断。
4. 向 EECON2 写入 55h。
5. 向 EECON2 写入 0AAh。
6. 将 WR 位置 1。这将开始行擦除周期。
7. CPU 在擦除期间将会停止工作 （使用内部定时器约为 2 ms）。
8. 重新允许中断。

例 6-2： 擦除闪存程序存储器的一行

```
            MOVLW  CODE_ADDR_UPPER        ; load TBLPTR with the base
            MOVWF  TBLPTRU                ; address of the memory block
            MOVLW  CODE_ADDR_HIGH
            MOVWF  TBLPTRH
            MOVLW  CODE_ADDR_LOW
            MOVWF  TBLPTRL
ERASE_ROW
            BSF    EECON1, WREN           ; enable write to memory
            BSF    EECON1, FREE           ; enable Row Erase operation
            BCF    INTCON, GIE            ; disable interrupts
            MOVLW  55h
            MOVWF  EECON2                 ; write 55h
必需的序列  MOVLW  0AAh
            MOVWF  EECON2                 ; write 0AAh
            BSF    EECON1, WR             ; start erase (CPU stall)
            BSF    INTCON, GIE            ; re-enable interrupts
```


6.5 写闪存程序存储器

最小编程块大小为 32 字或 64 字节。不支持字或字节编程。

在内部使用表写命令将需要写入闪存存储器的内容装入保持寄存器中。表写操作使用 64 个保持寄存器进行编程。

由于表锁存器 （TABLAT）只是单字节寄存器，所以每次编程操作， `TBLWT` 指令都必须执行 64 次。因为只写保持寄存器，所以所有的表写操作实际上都是短写。更新 64 个保持寄存器后，必须写 EECON1 寄存器，以便启动长写周期开始编程操作。

对内部闪存编程要求使用长写操作。在长写周期中，指令停止执行。由内部编程定时器终止长写操作。

由 EEPROM 片上定时器控制写入的时间。写入 / 擦除电压由片上的电荷泵产生，该电荷泵可以工作在器件的电压范围内。

> **注 1：** 与早期的 PIC 器件不同，PIC18F97J60 系列器件在写操作发生后并不会复位保持寄存器。必须在编程序列之前清零或改写保持寄存器。
>
> **2：** 为了保证程序存储器单元的耐擦写次数，在两次擦除操作之间不应对某一闪存字节编程多于一次。在第二次尝试改写目标行的内容之前，需要对其执行行擦除或整个存储器的批量擦除。

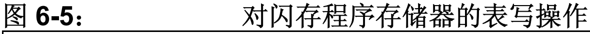
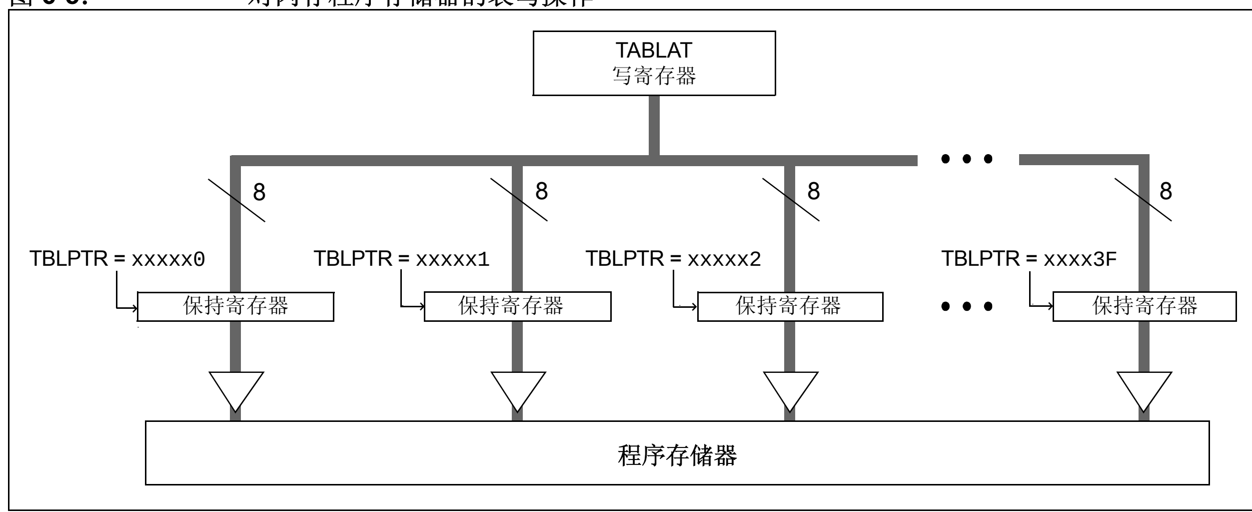


图 **6-5**： 对闪存程序存储器的表写操作



6.5.1 写入闪存程序存储器操作顺序

内部程序存储器单元的编程事件顺序应为：

1. 如果要写入的程序存储器段已被事先编程，那么在写入之前必须首先将其擦除 （见第 **6.4.1** 节 **"闪存程序存储器擦除序列"**）。
2. 通过自动递增将 64 个字节写入保持寄存器。
3. 将 WREN 位置 1 以使能字节写操作。
4. 禁止中断。
5. 向 EECON2 写入 55h。
6. 向 EECON2 写入 AAh。
7. 将 WR 位置 1。这将开始写周期。
8. CPU 在写入期间将会停止工作 （使用内部定时器约为 2 ms）。
9. 重新允许中断。
10. 验证存储器 （表读）。

例 6-3 给出了所需代码的示例。

> **注：** 在将 WR 位置 1 前，表指针必须指向保持寄存器中的 64 个字节的地址范围内。

例 6-3： 写闪存程序存储器

```
                MOVLW   CODE_ADDR_UPPER         ; Load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
ERASE_BLOCK
                BSF     EECON1, WREN            ; enable write to memory
                BSF     EECON1, FREE            ; enable Row Erase operation
                BCF     INTCON, GIE             ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2                  ; write 55h
                MOVLW   0AAh
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start erase (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
                MOVLW   D'16'
                MOVWF   WRITE_COUNTER           ; Need to write 16 blocks of 64 to write
                                                ; one erase block of 1024
RESTART_BUFFER
                MOVLW   D'64'
                MOVWF   COUNTER
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
FILL_BUFFER
                ...                             ; read the new data from I2C, SPI,
                                                ; PSP, USART, etc.
WRITE_BUFFER
                MOVLW   D’64                    ; number of bytes in holding register
                MOVWF   COUNTER
WRITE_BYTE_TO_HREGS
                MOVFF   POSTINC0, WREG          ; get low byte of buffer data
                MOVWF   TABLAT                  ; present data to table latch
                TBLWT+*                         ; write data, perform a short write
                                                ; to internal TBLWT holding register.
                DECFSZ  COUNTER                 ; loop until buffers are full
                BRA     WRITE_WORD_TO_HREGS
PROGRAM_MEMORY
                BSF     EECON1, WREN            ; enable write to memory
                BCF     INTCON, GIE             ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2                  ; write 55h
   必需的序列    MOVLW   0AAh
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start program (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
                BCF     EECON1, WREN            ; disable write to memory

                DECFSZ  WRITE_COUNTER           ; done with one write cycle
                BRA     RESTART_BUFFER          ; if not done replacing the erase block
```

 超前信息

6.5.2 写校验

根据具体应用情况，一个比较好的编程习惯可能要求将写入存储器的值与原始值作比较，进行校验。过多的写操作可能会导致某些位接近规范的极限，在这样的应用场合就应该采用写校验。

6.5.3 意外终止写操作

如果由于意外事件（如掉电或意外复位）终止了写操作，应该对刚刚编程的存储器单元进行验证，如有必要，还要重新进行编程。当写操作在正常操作过程中因 MCLR 复位或 WDT 超时复位而中断时，用户可以检查 WRERR 位，并根据需要重写该单元。

6.5.4 防止误写操作的保护措施

为防止对闪存程序存储器的误写操作，必须遵循写操作的启动顺序。更多详细信息，请参见**第 24.0 节 “CPU 的特殊功能”**。

6.6 代码保护期间闪存程序存储器的操作

关于闪存程序存储器代码保护的详细信息，请参见**第 24.6 节 “程序校验和代码保护”**。

表 6-2： 与闪存程序存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节（TBLPTR<20:16>）					59
TBPLTRH	程序存储器表指针高字节（TBLPTR<15:8>）								59
TBLPTRL	程序存储器表指针低字节（TBLPTR<7:0>）								59
TABLAT	程序存储器表锁存器								59
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
EECON2	EEPROM 控制寄存器 2（不是实际存在的寄存器）								61
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	61

图注： — = 未用，读为 0。闪存 /EEPROM 访问期间不使用阴影单元。

注：

7.0 外部存储器总线

注：	64 引脚和 80 引脚器件上未实现外部存储器总线。

外部存储器总线使器件可以访问外部存储器件 （如闪存、 EPROM、 SRAM 等），将其作为程序或数据存储器。它既支持 8 位和 16 位数据宽度模式，也支持三种地址宽度 （最高达 20 位）。

该总线与 4 个 I/O 端口复用，占 28 个引脚。有三个端口（PORTD、 PORTE 和 PORTH）与地址 / 数据总线复用，总共有 20 条线可供使用，而端口 PORTJ 和总线控制信号复用。

表 7-1 列出了各个引脚及其功能。

表 7-1： PIC18F96J60/96J65/97J60 外部总线——I/O 端口功能

名称	端口	位	外部存储器总线功能
RD0/AD0	PORTD	0	地址 bit 0 或数据 bit 0
RD1/AD1	PORTD	1	地址 bit 1 或数据 bit 1
RD2/AD2	PORTD	2	地址 bit 2 或数据 bit 2
RD3/AD3	PORTD	3	地址 bit 3 或数据 bit 3
RD4/AD4	PORTD	4	地址 bit 4 或数据 bit 4
RD5/AD5	PORTD	5	地址 bit 5 或数据 bit 5
RD6/AD6	PORTD	6	地址 bit 6 或数据 bit 6
RD7/AD7	PORTD	7	地址 bit 7 或数据 bit 7
RE0/AD8	PORTE	0	地址 bit 8 或数据 bit 8
RE1/AD9	PORTE	1	地址 bit 9 或数据 bit 9
RE2/AD10	PORTE	2	地址 bit 10 或数据 bit 10
RE3/AD11	PORTE	3	地址 bit 11 或数据 bit 11
RE4/AD12	PORTE	4	地址 bit 12 或数据 bit 12
RE5/AD13	PORTE	5	地址 bit 13 或数据 bit 13
RE6/AD14	PORTE	6	地址 bit 14 或数据 bit 14
RE7/AD15	PORTE	7	地址 bit 15 或数据 bit 15
RH0/A16	PORTH	0	地址 bit 16
RH1/A17	PORTH	1	地址 bit 17
RH2/A18	PORTH	2	地址 bit 18
RH3/A19	PORTH	3	地址 bit 19
RJ0/ALE	PORTJ	0	地址锁存使能 （ALE）控制引脚
RJ1/$\overline{\text{OE}}$	PORTJ	1	输出使能 （$\overline{\text{OE}}$）控制引脚
RJ2/$\overline{\text{WRL}}$	PORTJ	2	低字节写 （$\overline{\text{WRL}}$）控制引脚
RJ3/$\overline{\text{WRH}}$	PORTJ	3	高字节写 （$\overline{\text{WRH}}$）控制引脚
RJ4/BA0	PORTJ	4	字节地址 bit 0 （BA0）
RJ5/$\overline{\text{CE}}$	PORTJ	5	芯片使能 （$\overline{\text{CE}}$）控制引脚
RJ6/$\overline{\text{LB}}$	PORTJ	6	低字节使能 （$\overline{\text{LB}}$）控制引脚
RJ7/$\overline{\text{UB}}$	PORTJ	7	高字节使能 （$\overline{\text{UB}}$）控制引脚

注： 为清楚起见，此处仅显示了 I/O 端口和外部总线的分配情况。某些引脚上可能具有一个或多个其他复用功能。

7.1 外部存储器总线控制

接口的操作由 MEMCON 寄存器 （寄存器 7-1）控制。该寄存器可以在除单片机模式以外的所有程序存储器操作模式下使用。在单片机模式下该寄存器被禁止，且不能被写入。

EBDIS 位 （MEMCON<7>）控制总线操作及相关端口功能。清零 EBDIS 位将使能接口，并禁止端口的 I/O 功能及引脚上的其他复用功能。置 1 该位将使能 I/O 端口及其他功能，但允许接口在要求进行外部存储器操作时，改写引脚上的数据。在默认情况下，总是使能外部总线并禁止其他所有 I/O 功能。

EBDIS 位的操作还会受所使用的程序存储器模式影响。在第 **7.5** 节 **"程序存储器模式和外部存储器总线"** 中对此进行了更为详细的讨论。

WAIT 位允许为外部存储器操作插入等待状态。在第 **7.3** 节 **"等待状态"** 中对这些位的使用方法进行了讨论。

WM 位用于选择在总线工作于 16 位数据宽度模式下时使用的具体操作模式。在第 **7.6** 节 **"16 位数据宽度模式"** 中对这些操作模式进行了更为详细的讨论。选择 8 位数据宽度模式时 WM 位不起任何作用。

寄存器 7-1： MEMCON：外部存储器总线控制寄存器

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **EBDIS**：外部总线禁止位
- `1` = 当单片机访问外部存储器时将使能外部总线；否则，所有外部总线驱动器映射为 I/O 端口。
- `0` = 始终使能外部总线，并禁止 I/O 端口

bit 6 未用：读为 `0`

bit 5-4 **WAIT1:WAIT0**：表读和表写操作总线等待周期数位
- `11` = 表读和表写操作等待 0 个 TCY
- `10` = 表读和表写操作等待 1 个 TCY
- `01` = 表读和表写操作等待 2 个 TCY
- `00` = 表读和表写操作等待 3 个 TCY

bit 3-2 未用：读为 `0`

bit 1-0 **WM1:WM0**：16 位数据总线宽度下的 `TBLWT` 操作模式选择位
- `1x` = 字写模式：当写入 TABLAT 并且 TBLPTR 包含奇地址时，$\overline{WRH}$ 有效。当 TBLPTR 包含偶地址时，写入 TABLAT 会将写入值装入保持锁存器。
- `01` = 字节选择模式：TABLAT 数据复制到 MSB 和 LSB，$\overline{WRH}$ 与 $\overline{UB}$ （或 $\overline{LB}$）将有效
- `00` = 字节写模式：TABLAT 数据复制到 MSB 和 LSB， $\overline{WRH}$ 或 $\overline{WRL}$ 将有效

 超前信息

7.2 地址和数据宽度

PIC18F97J60 系列器件可以将同一存储器总线配置成不同的地址和数据宽度。地址和数据宽度都通过 CONFIG3L 寄存器中的配置位来设置。配置位意味着这些选项只能通过器件编程来配置，而不能通过软件进行控制。

BW 位是 8 位或 16 位数据总线宽度的选择位。置 1 该位 （默认）将选择 16 位数据宽度。

EMB1:EMB0 位同时决定程序存储器的操作模式和地址总线宽度。可用的选项有 20 位、16 位和 12 位地址总线宽度，以及默认的单片机模式 （禁止外部总线）。选择 16 位或 12 位将使多出的高位地址线可被用于 I/O 端口功能，这些引脚将不再受 EBDIS 位设置的影响。例如，选择 16 位地址模式（EMB1:EMB0 = 01）将禁止 A19:A16，并使 PORTH<3:0> 位可以不受总线影响而作为端口工作。使用较小的地址宽度使用户可以根据特定应用的外部存储器空间大小来定制存储器总线，同时又空出引脚专门用于 I/O操作。

由于 EMB 位可以禁止引脚用于存储器总线操作，因此要注意选择的地址宽度要始终大于等于数据宽度。如果为 16 位数据宽度配以 12 位地址宽度，则总线上将无法得到数据的高 4 位。

所有的地址与数据宽度组合都需要复用信号线。表 7-2 中对地址和数据线复用，以及使用较小地址宽度时可用的 I/O 端口进行了总结。

7.2.1 外部总线上的地址平移

在默认情况下，外部总线上的地址是 PC 的值。实际上，这意味着外部存储器器件中，位于片上存储器上边界以下的地址，对于单片机是不可用的。要访问这些物理单元，单片机与外部存储器之间的连接逻辑必须能够进行地址转换。

为了简化接口，外部总线提供了单片机模式，它可以自动执行地址平移。该功能由 EASHFT 配置位控制。置 1 该位会将总线上的地址进行偏移，偏移量为单片机片上程序存储器大小，且下限地址设为 0000h。这使器件可以使用外部存储器物理地址的全部范围。

7.2.2 21 位寻址

作为 20 位地址宽度操作的扩展，外部存储器总线还可以对 2 MB 的所有存储器空间进行寻址。这通过使用总线地址 Bit 0（BA0）控制线作为地址的最低有效位来实现。对于某些存储器器件，还可以使用 UB 和 LB 控制信号来选择 16 位宽数据字中的高字节和低字节。

在 8 位数据宽度和某些 16 位数据宽度模式中可以使用该寻址模式。更多详细信息，请参见**第 7.6.3 节 “16 位字节选择模式”**和**第 7.7 节 “8 位数据宽度模式”**。

表 7-2： 不同地址和数据宽度的地址和数据线

数据宽度	地址宽度	数据和地址复用线（及相应的端口）	仅地址线（及相应的端口）	可用于 I/O 的端口
8 位	12 位	AD7:AD0（PORTD<7:0>）	AD11:AD8（PORTE<3:0>）	PORTE<7:4>，PORTH 的所有位
	16 位		AD15:AD8（PORTE<7:0>）	PORTH 的所有位
	20 位		A19:A16 和 AD15:AD8（PORTH<3:0>，PORTE<7:0>）	—
16 位	16 位	AD15:AD0（PORTD<7:0>，PORTE<7:0>）	—	PORTH 的所有位
	20 位		A19:A16（PORTH<3:0>）	—

7.3 等待状态

人们可能认为外部存储器件是工作在单片机的时钟速率下的，但通常并非如此。事实上，许多器件写数据或检索数据所需的时间要比执行表读或表写操作所允许的时间长。

为补偿这一点，可以将外部存储器总线配置为在每个使用总线的表操作中增加固定的延时。等待状态可通过将 WAIT 配置位置 1 来使能。使能后，延时时间量由 WAIT1:WAIT0 位（MEMCON<5:4>）设置。该延时为单片机指令周期的整数倍，并在执行表操作时添加到其指令周期后面。范围从无延时到 3 个 TCY（默认值）。

7.4 端口引脚弱上拉

除了高地址线 A19:A16，与外部存储器总线相关的引脚都配有弱上拉电路。上拉功能由 PORTG 寄存器的高 3 位控制。它们是 RDPU、REPU 和 RJPU，分别控制 PORTD、PORTE 和 PORTJ 端口的弱上拉。置 1 其中任何一位将使能相应端口的弱上拉。默认情况下，器件复位时所有弱上拉都被禁止。

7.5 程序存储器模式和外部存储器总线

使用片上程序存储器和外部程序存储器组合，可使 PIC18F97J60 系列器件能够工作于两种程序存储器模式下。复用端口引脚的功能取决于所选择的程序存储器模式和 EBDIS 位的设置。

在**单片机模式**下，总线处于无效状态，引脚只具有端口功能。写 MEMCOM 寄存器是不允许的。EBDIS 的复位值（0）被忽略， EMB 引脚用作 I/O 端口。

在**扩展单片机模式**下，外部程序存储器总线共用引脚上的 I/O 端口功能。当器件在外部程序存储空间上取数据或进行表读 / 表写操作时，引脚将具有外部总线功能。

如果器件仅对内部程序存储器取数据和进行访问，EBDIS 控制位将把引脚从外部存储器功能变换为 I/O 端口功能。当 EBDIS = 0 时，引脚用作外部总线。当 EBDIS = 1 时，引脚用作 I/O 端口。

如果在 EBDIS = 1 时，器件对外部存储器取数据或进行访问，引脚将转换为外部总线。如果外部存储器执行的程序将 EBDIS 位置 1，则置 1 该位的操作将被延迟，直到程序跳转到内部存储器。此时，引脚将从外部总线变换为 I/O 端口。

如果在 EBDIS = 0 时，器件执行内部存储器操作，存储器总线地址 / 数据引脚和控制引脚将无效。它们将转为以下状态：有效地址 / 数据引脚为三态， $\overline{CE}$、$\overline{OE}$、$\overline{WRH}$、$\overline{WRL}$、$\overline{UB}$ 和 $\overline{LB}$ 信号为 1，而 ALE 和 BA0 为 0。请注意，只有那些与当前地址宽度相关的引脚才会被强制设为三态，其他引脚将继续用作 I/O 端口。例如，在地址为 16 位宽时，只有 AD<15:0>（PORTD 和 PORTE）会受影响，A19:A16（PORTH<3:0>）继续用作 I/O 端口。

在所有外部存储器模式中，总线的优先级高于其他任何与其共用引脚的外设。其中包括并行从动端口（PSP）和串行通信模块；在非外部存储器模式时，它们的优先级高于 I/O 端口。

7.6 16 位数据宽度模式

在 16 位数据宽度模式中，外部存储器接口可以使用三种不同的配置与外部存储器连接：

- 16 位字节写
- 16 位字写
- 16 位字节选择

要使用的配置取决于 MEMCON 寄存器中的 WM1:WM0（MEMCON<1:0>）位。这三种不同配置使设计者可以在使用8位和16位器件处理16位数据时获得最大的灵活性。

对于所有 16 位数据宽度模式，地址锁存使能（ALE）引脚用于表明地址位 AD<15:0> 在外部存储器接口总线上是有效的。地址锁存之后，输出使能信号（$\overline{OE}$）将一次使能程序存储器的两个字节来形成一个 16 位的指令字。芯片使能信号（$\overline{CE}$）在单片机访问外部存储器的任何时刻，无论是执行读操作还是写操作时，都是有效的。每当器件处于休眠模式时，它是无效的（保持高电平）。

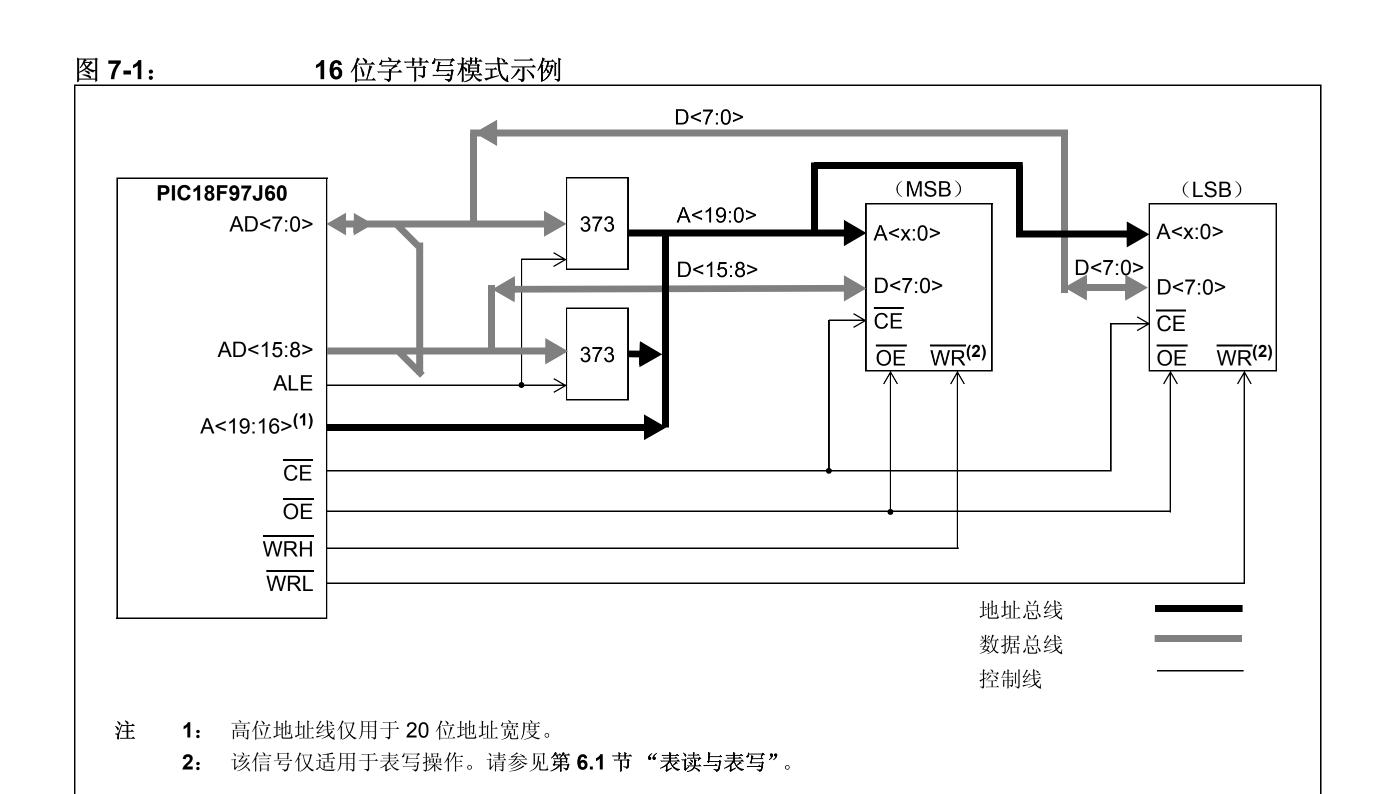
在字节选择模式中， JEDEC 标准闪存存储器需要使用 BA0 作为字节地址线，并使用一条 I/O 线来选择字节或字模式。其他 16 位数据宽度模式不需要使用 BA0。JEDEC 标准静态 RAM 存储器将使用 $\overline{UB}$ 或 $\overline{LB}$ 信号来选择字节。

7.6.1 16 位字节写模式

图 7-1 给出了 PIC18F97J60 系列器件 16 位字节写模式的一个示例。该模式用于连接两个独立 8 位存储器进行 16 位操作。这通常包括基本 EPROM 和闪存器件。它允许对字节宽度的外部存储器进行表写操作。

在TBLWT指令周期中，TABLAT中的数据送到AD15:AD0总线的高字节和低字节。TBLPTR 的最低有效位将选通相应的 WRH 或 WRL 控制线。

图 7-1： 16 位字节写模式示例



注 1： 高位地址线仅用于 20 位地址宽度。

2： 该信号仅适用于表写操作。请参见**第 6.1 节 “表读与表写”**。

7.6.2 16 位字写模式

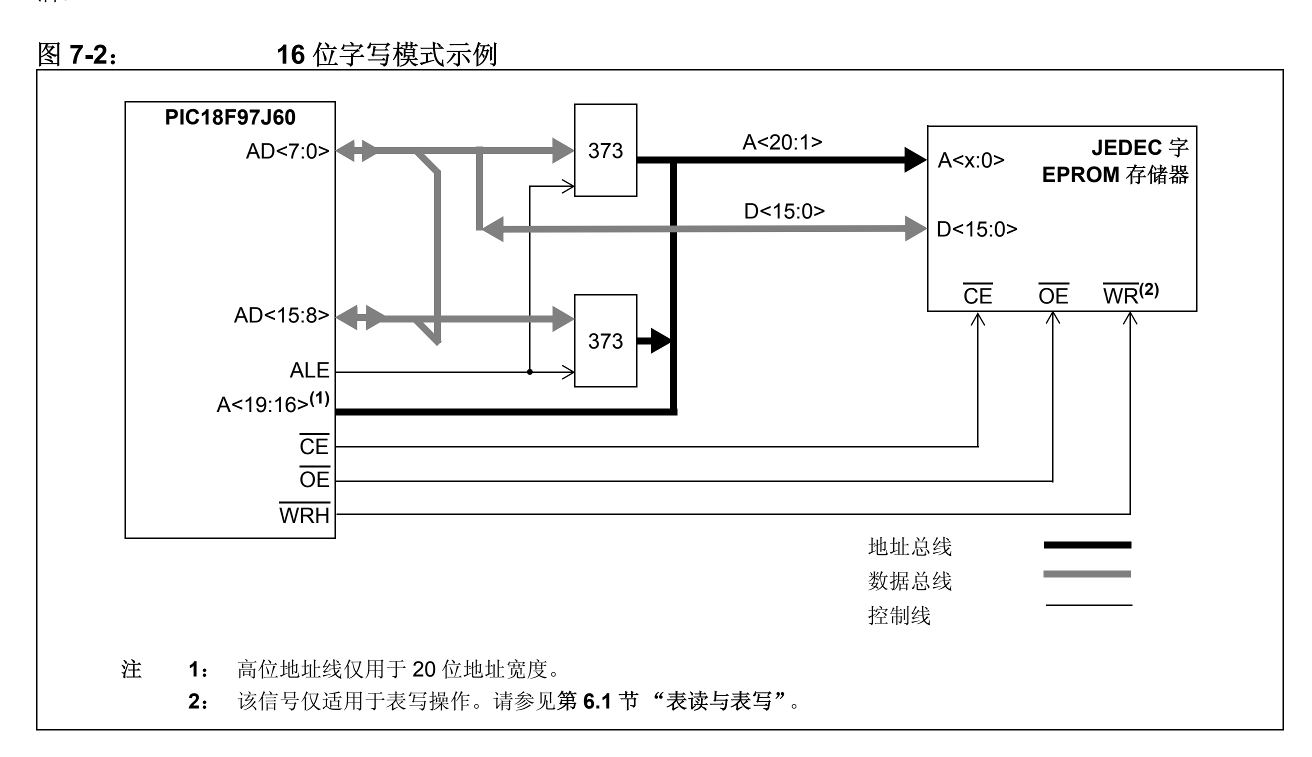
图 7-2 给出了 PIC18F97J60 器件 16 位字写模式的一个示例。该模式用于以字长为单位的存储器，包括一些 EPROM 和闪存类型的存储器。该模式允许从所有形式的 16 位存储器中取操作码和对其进行表读操作，以及对任意类型的 16 位外部存储器进行表写操作。使用该方法时，访问奇地址和偶地址的 TBLWT 周期之间存在区别。

对于访问偶地址的 TBLWT 周期（TBLPTR<0> = 0），TABLAT 中的数据将传送到保持锁存器，外部地址数据总线在总线作为数据总线时被置为三态。写信号不会激活。

对于访问奇地址的 TBLWT 周期（TBLPTR<0> = 1），TABLAT 中的数据送到 AD15:AD0 总线的高字节。保持锁存器的内容送到 AD15:AD0 总线的低字节。

$\overline{\text{WRH}}$信号在每个写周期被选通，$\overline{\text{WRL}}$引脚未使用。BA0 引脚上的信号表明 TBLPTR 的最低有效位，但处于未连接状态。实际上将使用 $\overline{\text{UB}}$ 和 $\overline{\text{LB}}$ 信号来选择两个字节。该方法的明显局限在于，必须在特定字边界成对地执行表写操作，才能正确地写一个字单元。

图 7-2： 16 位字写模式示例



注 1： 高位地址线仅用于 20 位地址宽度。
2： 该信号仅适用于表写操作。请参见第 6.1 节 “表读与表写”。

7.6.3 16 位字节选择模式

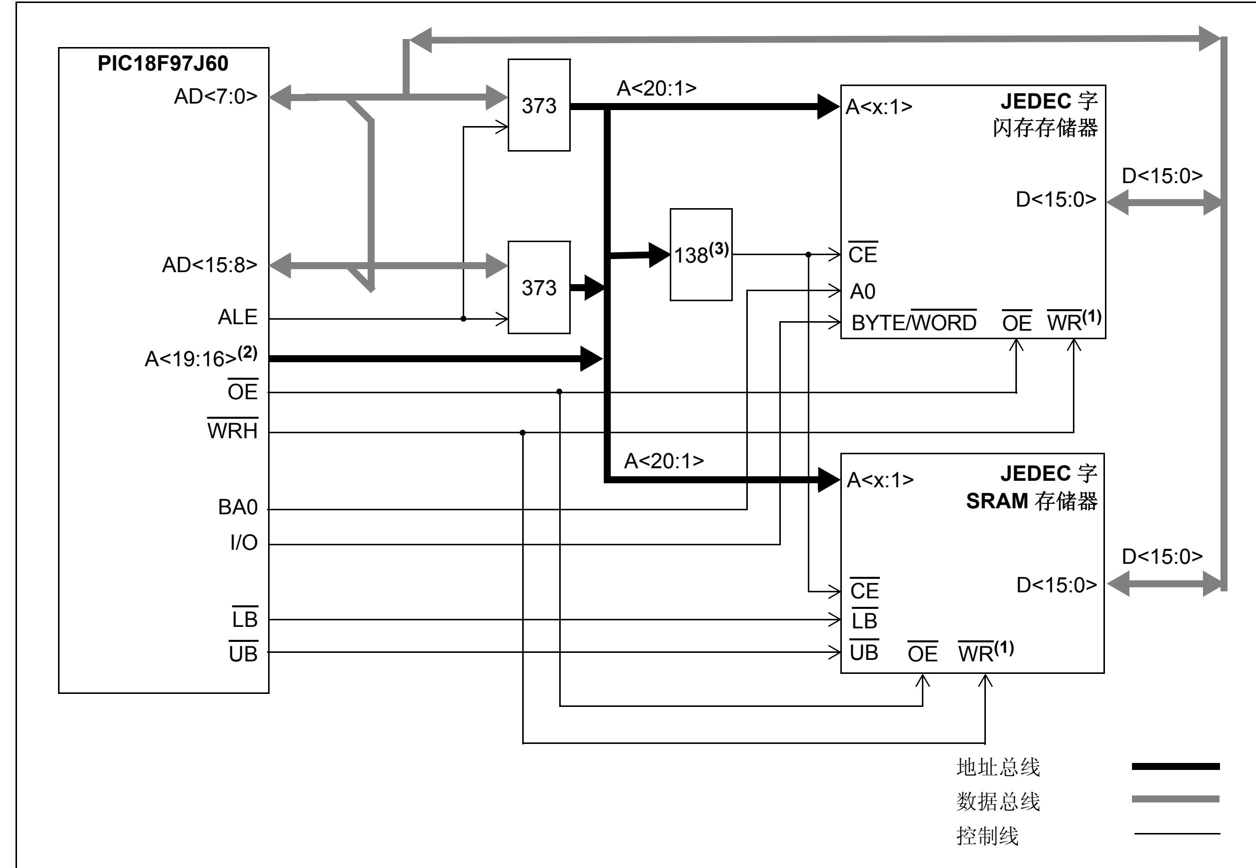
图 7-3 给出了 16 位字节选择模式的一个示例。该模式允许向字宽度的外部存储器进行表写操作，同时还可以进行字节选择。这通常包括字宽度的闪存和 SRAM 器件。

在 TBLWT 周期中，TABLAT 中的数据送到 AD15:AD0 总线的高字节和低字节。WRH 信号在每一个写周期被选通，WRL 引脚未使用。BA0 或 UB/LB 信号根据 TBLPTR 寄存器的最低有效位选择要写的字节。

闪存和SRAM器件使用不同的控制信号组合来实现字节选择模式。JEDEC 标准闪存存储器要求将一个控制器 I/O 端口引脚连接到存储器的 BYTE/WORD 引脚来提供选择信号。它们使用控制器的 BA0 信号作为字节地址。而 JEDEC 标准静态 RAM 存储器则使用 UB 或 LB 信号来选择字节。



图 7-3： 16 位字节选择模式示例



注 1： 该信号仅适用于表写操作。请参见**第 6.1 节 “表读与表写”**。
2： 高位地址线仅用于 20 位地址宽度。
3： 只有在访问多个存储器件时才需要进行信号分离。

7.6.4 16 位模式时序

对于各种操作模式，送到外部存储器总线的控制信号是不同的。图 7-4 和图 7-5 给出了典型的信号时序图。

图 7-4： 执行 TBLRD 操作时外部存储器总线的时序 （扩展单片机模式）

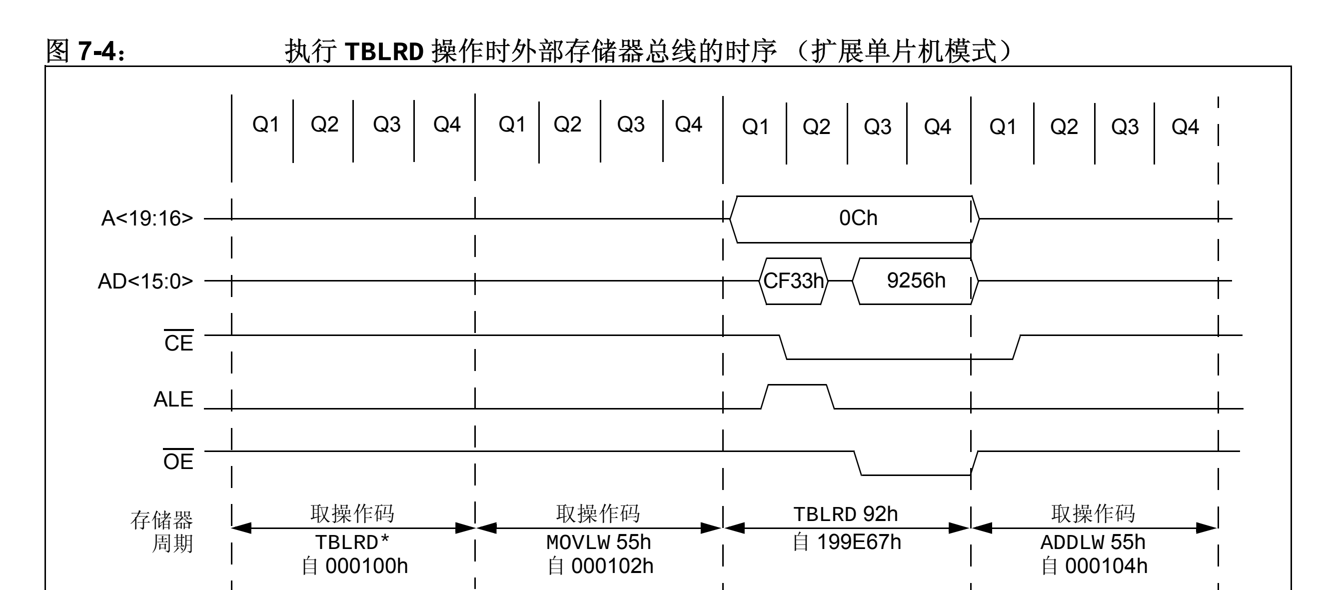
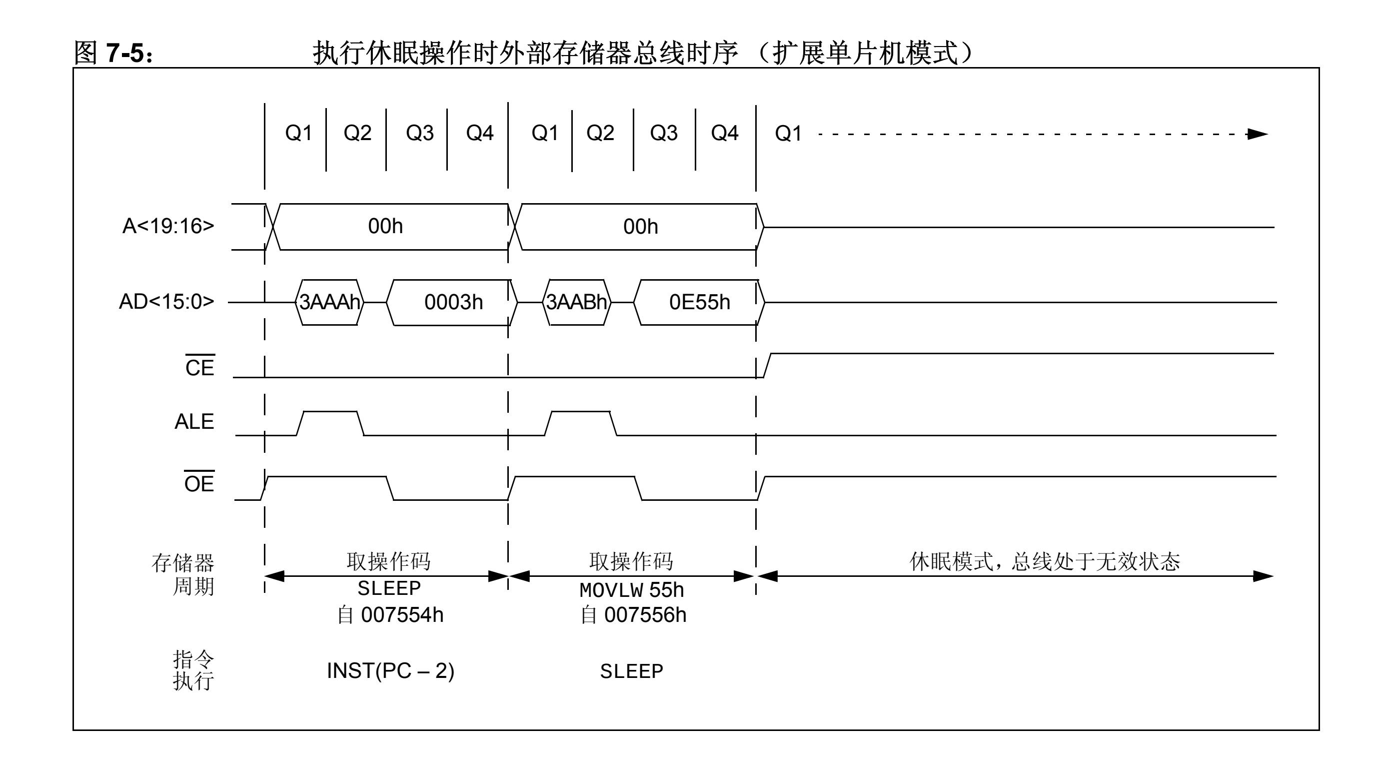


图 7-5： 执行休眠操作时外部存储器总线时序 （扩展单片机模式）



 超前信息

7.7 8 位数据宽度模式

在 8 位数据宽度模式中，外部存储器总线仅工作于复用模式；也就是说，数据共用地址总线的低 8 位。

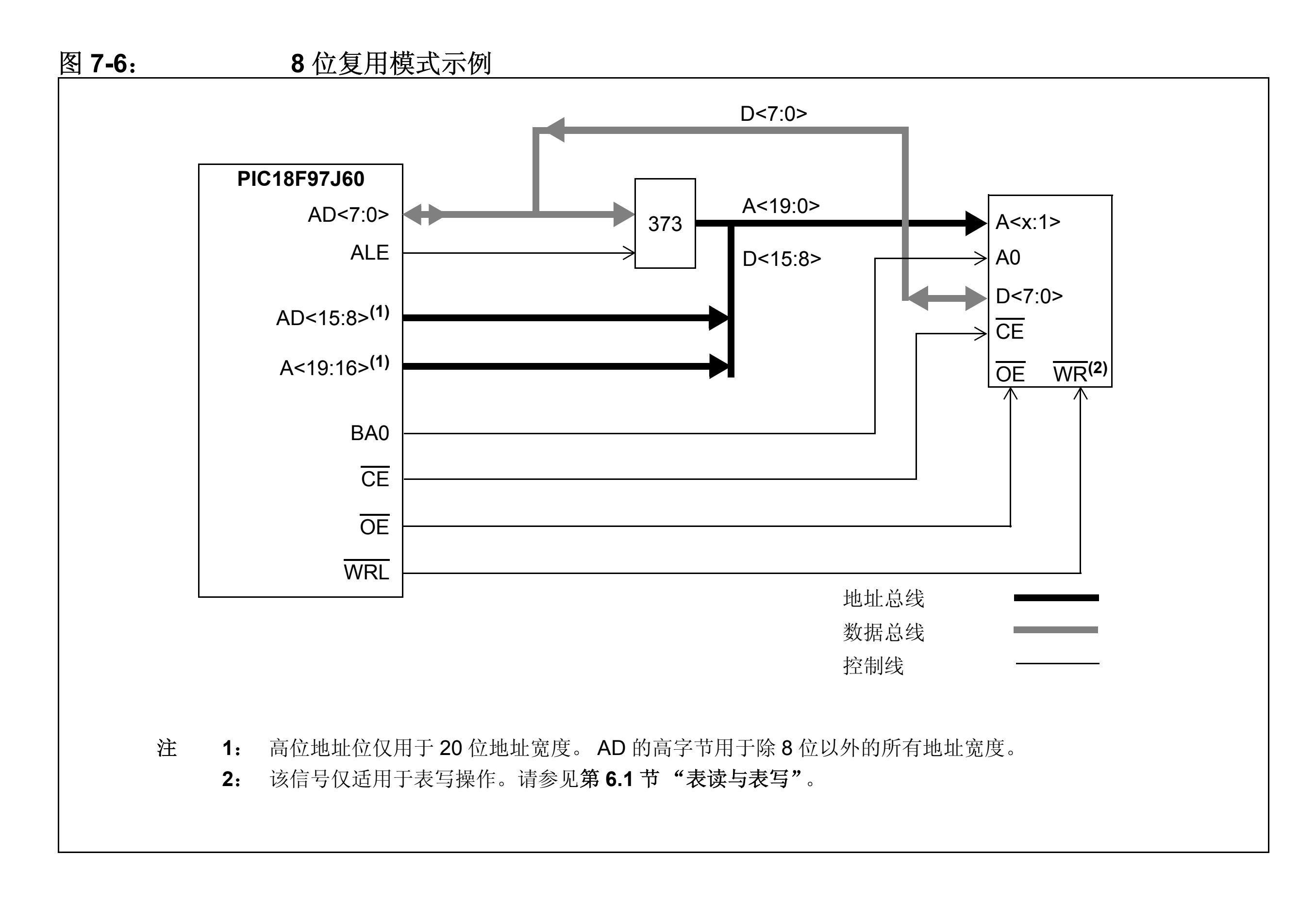
图 7-6 给出了 100 引脚器件 8 位复用模式的一个示例。该模式用于对单个 8 位存储器进行 16 位操作。在数据 / 地址复用总线上，将按照取两个 8 位字节的方式取指令。这两个字节在一个指令周期（TCY）内顺序地取出。因此，设计者必须以 1/2 TCY（双倍指令速率）为单位计算时序，并据此来选择外部存储器。为了选择合适的存储器速率，必须在考虑建立时间和保持时间的同时，考虑连接逻辑传输延时。

地址锁存使能（ALE）引脚表明地址位 AD<15:0> 在外部存储器接口总线上是有效的。输出使能信号（$\overline{\text{OE}}$）将在指令周期的部分时间内使能程序存储器的一个字节，然后 BA0 将改变，从而使能第二个字节来形成 16 位指令字。在该模式下，地址的最低有效位 BA0 必须连接到存储器件。芯片使能信号（$\overline{\text{CE}}$）在单片机访问外部存储器的任何时刻，无论是执行读操作还是写操作时，都是有效的。每当器件处于休眠模式时，它是无效的（保持高电平）。

该过程通常包括基本 EPROM 和闪存器件。它允许对字节宽度的外部存储器进行表写操作。

在 TBLWT 指令周期中，TABLAT 中的数据送到 AD15:AD0 总线的高字节和低字节。在 TBLPTR 的最低有效位将选通 BA0 控制线，送出适当的电平。

图 7-6： 8 位复用模式示例



注 1： 高位地址位仅用于 20 位地址宽度。AD 的高字节用于除 8 位以外的所有地址宽度。

2： 该信号仅适用于表写操作。请参见**第 6.1 节 “表读与表写”**。

7.7.1 8 位模式时序

对于各种操作模式，送到外部存储器总线的控制信号是不同的。图 7-7 和图 7-8 给出了典型的信号时序图。

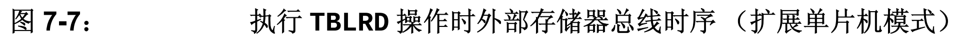


图 7-7： 执行 TBLRD 操作时外部存储器总线时序 （扩展单片机模式）

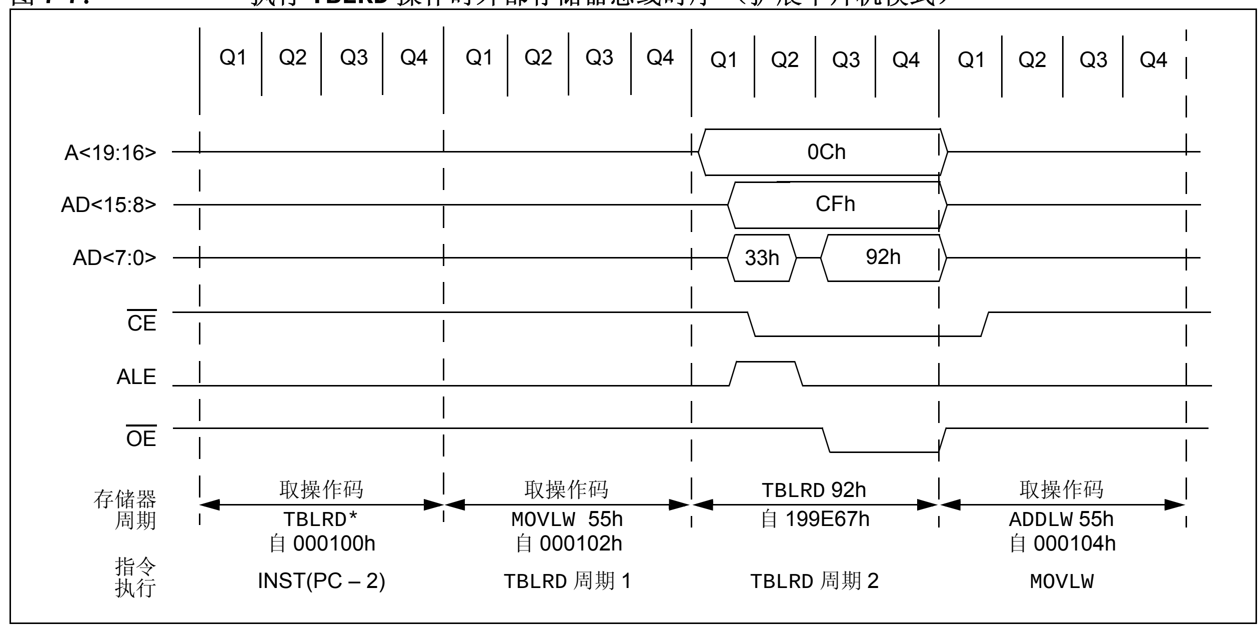
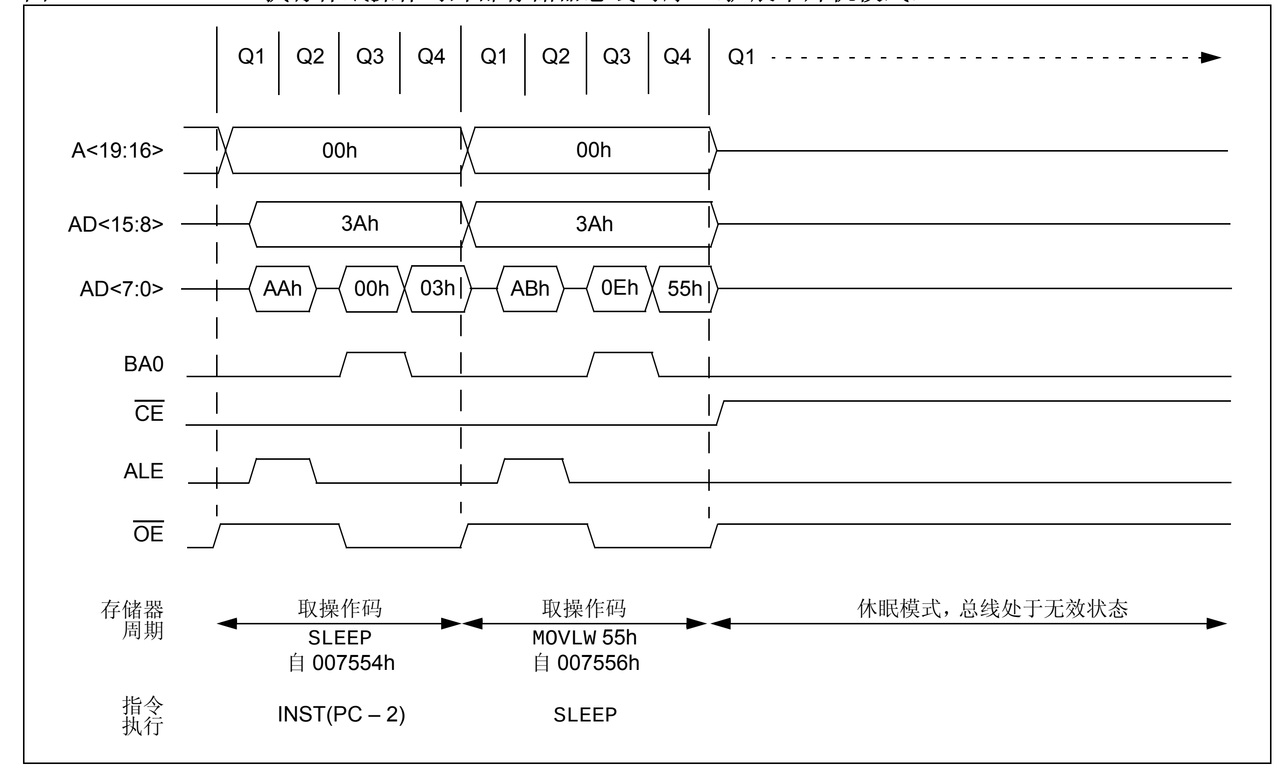


图 7-8： 执行休眠操作时外部存储器总线时序 （扩展单片机模式）



 超前信息

7.8 功耗管理模式下的操作

在功耗管理运行模式下，外部存储器总线继续正常工作。如果选择了一个较低速的时钟源，总线操作以该速率运行。在这些情况下，如果使能等待状态且在外部存储器操作中增加了等待状态，则可能造成外部存储器访问时间过长。如果希望以低功耗运行模式执行操作，则用户应用程序应在低时钟速率时对存储器访问时间进行调整。

在休眠和空闲模式下，单片机内核不需要访问数据，总线操作会暂停。外部总线的状态被冻结，地址 / 数据引脚和大部分控制引脚保持在模式激活时的状态。唯一可能变化的是 $\overline{CE}$、$\overline{LB}$ 和 $\overline{UB}$ 引脚，它们被保持为逻辑高电平。

注：

8.0 8 x 8 硬件乘法器

8.1 简介

所有 PIC18 器件均包含一个 8 x 8 硬件乘法器（是 ALU 的一部分）。该乘法器可执行无符号运算并产生一个 16 位运算结果，该结果存储在一对乘积寄存器 PRODH:PRODL 中。该乘法器执行的运算不会影响 STATUS 寄存器中的任何标志。

通过硬件执行乘法运算只需要一个指令周期。硬件乘法器具有更高的计算吞吐量并减少了乘法算法的代码长度，从而可在许多先前仅能使用数字信号处理器的应用中使用 PIC18 器件。表 8-1 给出了硬件和软件乘法运算的比较，包括所需存储器空间和执行时间。

8.2 工作原理

例 8-1 给出了一个 8 x 8 无符号乘法运算的指令序列。当已在 WREG 寄存器中装入了一个乘数时，实现该运算仅需一条指令。

例 8-2 给出了一个 8 x 8 有符号乘法运算的指令序列。要弄清乘数的符号位，必须检查每个乘数的最高有效位（MSb），并做相应的减法。

例 8-1： 8 x 8 无符号乘法程序

```
MOVF    ARG1, W     ;
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
```

例 8-2： 8 x 8 有符号乘法程序

```
MOVF    ARG1, W
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
BTFSC   ARG2, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG2
```

表 8-1： 各种乘法运算的性能比较

程序	乘法实现方法	程序存储器（字）	周期数（最多）	时间		
				40 MHz 时	**10 MHz 时**	**4 MHz 时**
8 x 8 无符号	无硬件乘法	13	69	6.9 μs	27.6 μs	69 μs
	硬件乘法	1	1	100 ns	400 ns	1 μs
8 x 8 有符号	无硬件乘法	33	91	9.1 μs	36.4 μs	91 μs
	硬件乘法	6	6	600 ns	2.4 μs	6 μs
16 x 16 无符号	无硬件乘法	21	242	24.2 μs	96.8 μs	242 μs
	硬件乘法	28	28	2.8 μs	11.2 μs	28 μs
16 x 16 有符号	无硬件乘法	52	254	25.4 μs	102.6 μs	254 μs
	硬件乘法	35	40	4.0 μs	16.0 μs	40 μs

例 8-3 给出了一个 16 x 16 无符号乘法运算的指令序列。公式 8-1 为所使用的算法。32 位结果存储在 4 个寄存器（RES3:RES0）中。

公式 8-1：　　16 x 16 无符号乘法算法

$$
\begin{aligned}
RES3{:}RES0 &= ARG1H{:}ARG1L \bullet ARG2H{:}ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

例 8-3：　　16 x 16 无符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

例 8-4 给出了 16 x 16 有符号乘法运算的指令序列。公式 8-2 为所使用的算法。32 位结果存储在 4 个寄存器（RES3:RES0）中。要弄清乘数的符号位，必须检查每个乘数的最高有效位 （MSb），并做相应的减法。

公式 8-2：　　16 x 16 有符号乘法算法

$$
\begin{aligned}
RES3{:}RES0 &= ARG1H{:}ARG1L \bullet ARG2H{:}ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H\langle 7\rangle \bullet ARG1H{:}ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H\langle 7\rangle \bullet ARG2H{:}ARG2L \bullet 2^{16})
\end{aligned}
$$

例 8-4：　　16 x 16 有符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    BTFSS   ARG2H, 7        ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1       ; no, check ARG1
    MOVF    ARG1L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG1H, W        ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7        ; ARG1H:ARG1L neg?
    BRA     CONT_CODE       ; no, done
    MOVF    ARG2L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG2H, W        ;
    SUBWFB  RES3
;
CONT_CODE
   :
```

9.0 中断

PIC18F97J60 系列器件具有多个中断源及一个中断优先级功能，该功能可以给大多数中断源分配高优先级或者低优先级。高优先级中断向量位于 0008h，低优先级中断向量位于 0018h。高优先级中断事件可以中断正在处理的低优先级中断。

有 13 个寄存器用于控制中断操作。这些寄存器是：

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1、 PIR2 和 PIR3
- PIE1、 PIE2 和 PIE3
- IPR1、 IPR2 和 IPR3

建议使用 MPLAB® IDE 提供的 Microchip 头文件命名这些寄存器中的位。这使得汇编器 / 编译器能够自动识别指定寄存器内的这些位。

通常，中断源有 3 个位用于控制其操作。这些位的功能分别是：

- **标志位**表明发生了中断事件
- **允许位**允许程序跳转到中断向量地址处执行 （当标志位置 1 时）
- **优先级位**用于选择高优先级还是低优先级

通过将 IPEN 位 （RCON<7>）置 1，可使能中断优先级功能。当使能中断优先级时，有 2 个全局中断允许位。将 GIEH 位 （INTCON<7>）置 1，可允许所有优先级位已置 1（高优先级）的中断。将 GIEL 位（INTCON<6>）置 1，可允许所有优先级位已清零（低优先级）的中断。当中断标志位、允许位及相应的全局中断允许位均被置 1 时，中断将根据设置的中断优先级立即跳转到地址 0008h 或 0018h。也可以通过设置相应的允许位来禁止单个中断。

当 IPEN 位清零 （默认状态）时，便会禁止中断优先级功能，此时中断是与 PIC® 中档系列器件兼容的。在兼容模式下，各个中断源的中断优先级位不起作用。INTCON<6> 是 PEIE 位，用于允许 / 禁止所有的外设中断源。 INTCON<7> 是 GIE 位，用于允许 / 禁止所有中断源。在兼容模式下，所有中断均跳转到 0008h。

当响应中断时，全局中断允许位被清零以禁止其他中断。如果 IPEN 位是零，则该位就是 GIE 位。如果使用了中断优先级，这个位就是 GIEH 位或者 GIEL 位。高优先级中断源会中断低优先级中断。在处理高优先级中断时，低优先级中断将不被响应。

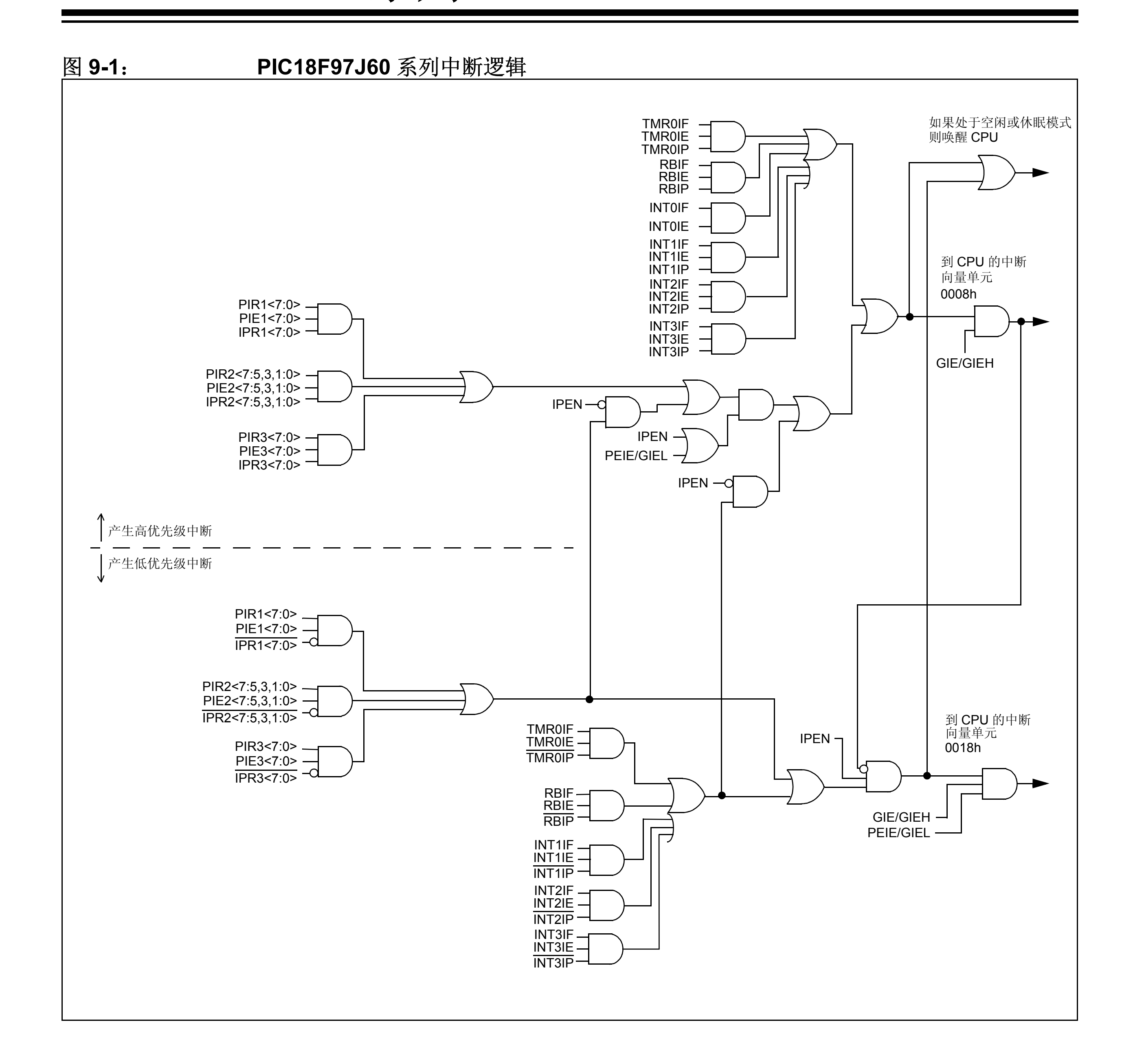
返回地址被压入堆栈，中断向量地址（0008h或0018h）被装入 PC。只要在中断服务程序中，就可以通过查询中断标志位来确定中断源。在重新允许中断前，必须用软件将中断标志位清零，以避免重复响应中断。

执行 “从中断返回”指令 `RETFIE` 将退出中断程序，同时将 GIE 位（若使用中断优先级则为 GIEH 或 GIEL 位）置 1，从而重新允许中断。

对于外部中断事件，例如 INT 引脚中断或者 PORTB 输入电平变化中断，中断响应延时将会是 3 到 4 个指令周期。对于单周期或双周期指令，中断响应延时完全相同。各中断标志位的置 1 不受对应的中断允许位和 GIE 位状态的影响。

注：	当允许**任何**中断时，不要使用 `MOVFF` 指令修改中断控制寄存器。否则可能导致单片机操作出错。

图 9-1：　　PIC18F97J60 系列中断逻辑



9.1 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含多个允许位、优先级位和标志位。

注：	当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断前，先将相应的中断标志位清零。中断标志位可由软件查询。

寄存器 9-1： INTCON：中断控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF[(1)]
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **GIE/GIEH**：全局中断允许位
当 IPEN = 0 时：
1 = 允许所有未被屏蔽的中断
0 = 禁止所有中断
当 IPEN = 1 时：
1 = 允许所有高优先级中断
0 = 禁止所有中断

bit 6 **PEIE/GIEL**：外设中断允许位
当 IPEN = 0 时：
1 = 允许所有未被屏蔽的外设中断
0 = 禁止所有外设中断
当 IPEN = 1 时：
1 = 允许所有低优先级的外设中断
0 = 禁止所有低优先级的外设中断

bit 5 **TMR0IE**：TMR0 溢出中断允许位
1 = 允许 TMR0 溢出中断
0 = 禁止 TMR0 溢出中断

bit 4 **INT0IE**：INT0 外部中断允许位
1 = 允许 INT0 外部中断
0 = 禁止 INT0 外部中断

bit 3 **RBIE**：RB 端口电平变化中断允许位
1 = 允许 RB 端口电平变化中断
0 = 禁止 RB 端口电平变化中断

bit 2 **TMR0IF**：TMR0 溢出中断标志位
1 = TMR0 寄存器已溢出 （必须用软件清零）
0 = TMR0 寄存器未溢出

bit 1 **INT0IF**：INT0 外部中断标志位
1 = 发生了 INT0 外部中断 （必须用软件清零）
0 = 未发生 INT0 外部中断

bit 0 **RBIF**：RB 端口电平变化中断标志位 [(1)]
1 = RB7:RB4 引脚中至少有一个引脚的电平状态发生了改变 （必须用软件清零）
0 = RB7:RB4 引脚电平状态没有改变

注 1： 电平的不匹配会不断地将该位置 1。读取 PORTB 可以结束不匹配情况，并将该位清零。

寄存器 9-2： INTCON2：中断控制寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **$\overline{\text{RBPU}}$**：PORTB 上拉使能位
1 = 禁止所有 PORTB 上拉
0 = 根据各端口锁存值使能 PORTB 上拉

bit 6 **INTEDG0**：外部中断 0 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 5 **INTEDG1**：外部中断 1 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 4 **INTEDG2**：外部中断 2 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 3 **INTEDG3**：外部中断 3 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 2 **TMR0IP**：TMR0 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **INT3IP**：INT3 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **RBIP**：RB 端口电平变化中断优先级位
1 = 高优先级
0 = 低优先级

注： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断前，先将相应的中断标志位清零。中断标志位可由软件查询。

寄存器 9-3： INTCON3：中断控制寄存器 3

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **INT2IP**：INT2 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **INT1IP**：INT1 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **INT3IE**：INT3 外部中断允许位
1 = 允许 INT3 外部中断
0 = 禁止 INT3 外部中断

bit 4 **INT2IE**：INT2 外部中断允许位
1 = 允许 INT2 外部中断
0 = 禁止 INT2 外部中断

bit 3 **INT1IE**：INT1 外部中断允许位
1 = 允许 INT1 外部中断
0 = 禁止 INT1 外部中断

bit 2 **INT3IF**：INT3 外部中断标志位
1 = 发生了 INT3 外部中断 （必须用软件清零）
0 = 未发生 INT3 外部中断

bit 1 **INT2IF**：INT2 外部中断标志位
1 = 发生了 INT2 外部中断 （必须用软件清零）
0 = 未发生 INT2 外部中断

bit 0 **INT1IF**：INT1 外部中断标志位
1 = 发生了 INT1 外部中断 （必须用软件清零）
0 = 未发生 INT1 外部中断

注：	当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断前，先将相应的中断标志位清零。中断标志位可由软件查询。

9.2 PIR 寄存器

PIR 寄存器包含各外设中断的标志位。根据外设中断源的数量，有三个外设中断请求（标志）寄存器（PIR1、PIR2 和 PIR3）。

> 注 1：当中断条件产生时，不管相应的中断允许位或全局中断允许位 GIE（INTCON<7>）的状态如何，中断标志位都将置 1。
>
> 2: 用户软件应在允许一个中断前和处理完一次中断后，将相应的中断标志位清零。

寄存器 9-4： PIR1：外设中断请求 （标志）寄存器 1

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF(1)	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **PSPIF**：并行从动端口读 / 写中断标志位 (1)
1 = 发生了读或写操作 （必须用软件清零）
0 = 未发生读或写操作

bit 6 **ADIF**：A/D 转换器中断标志位
1 = 一次 A/D 转换已完成 （必须用软件清零）
0 = A/D 转换未完成

bit 5 **RC1IF**：EUSART1 接收中断标志位
1 = EUSART1 接收缓冲区 RCREG1 已满 （读取 RCREG1 时清零）
0 = EUSART1 接收缓冲区为空

bit 4 **TX1IF**：EUSART1 发送中断标志位
1 = EUSART1 发送缓冲区 TXREG1 为空 （写入 TXREG1 时清零）
0 = EUSART1 发送缓冲区已满

bit 3 **SSP1IF**：主控同步串口 1 中断标志位
1 = 发送 / 接收已完成 （必须用软件清零）
0 = 等待发送 / 接收

bit 2 **CCP1IF**：ECCP1 中断标志位
捕捉模式：
1 = 发生了 TMR1 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1 寄存器捕捉
比较模式：
1 = 发生了 TMR1 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

bit 1 **TMR2IF**：TMR2 与 PR2 匹配中断标志位
1 = TMR2 与 PR2 发生匹配 （必须用软件清零）
0 = TMR2 与 PR2 未发生匹配

bit 0 **TMR1IF**：TMR1 溢出中断标志位
1 = TMR1 寄存器已溢出 （必须用软件清零）
0 = TMR1 寄存器未溢出

注 1： 仅在单片机模式下的 100 引脚器件上实现。

寄存器 9-5： **PIR2：外设中断请求 （标志）寄存器 2**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **OSCFIF**：振荡器失效中断标志位
1 = 系统振荡器失效，改成由 INTRC 作为时钟输入 （必须用软件清零）
0 = 系统时钟正常运行

bit 6 **CMIF**：比较器中断标志位
1 = 比较器输入已改变 （必须用软件清零）
0 = 比较器输入未变化

bit 5 **ETHIF**：以太网模块中断标志位
1 = 发生了以太网模块中断事件；查询 EIR 寄存器来判断中断原因
0 = 未发生以太网模块中断事件

bit 4 保留：保持为 0

bit 3 **BCL1IF**：总线冲突中断标志位 （MSSP1 模块）
1 = 发生了总线冲突 （必须用软件清零）
0 = 未发生总线冲突

bit 2 未用：读为 0

bit 1 **TMR3IF**：TMR3 溢出中断标志位
1 = TMR3 寄存器已溢出 （必须用软件清零）
0 = TMR3 寄存器未溢出

bit 0 **CCP2IF**：ECCP2 中断标志位
捕捉模式：
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
比较模式：
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

寄存器 9-6： PIR3：外设中断请求 （标志）寄存器 3

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IF(1)	BCL2IF(1)	RC2IF(2)	TX2IF(2)	TMR4IF	CCP5IF	CCP4IF	CCP3IF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SSP2IF**：主控同步串口 2 中断标志位 [1]
1 = 发送 / 接收已完成 （必须用软件清零）
0 = 等待发送 / 接收

bit 6 **BCL2IF**：总线冲突中断标志位 （MSSP2 模块） [1]
1 = 发生了总线冲突 （必须用软件清零）
0 = 未发生总线冲突

bit 5 **RC2IF**：EUSART2 接收中断标志位 [2]
1 = EUSART2 接收缓冲区 RCREG2 已满 （读取 RCREG2 时清零）
0 = EUSART2 接收缓冲区为空

bit 4 **TX2IF**：EUSART2 发送中断标志位 [2]
1 = EUSART2 发送缓冲区 TXREG2 为空 （写入 TXREG2 时清零）
0 = EUSART2 发送缓冲区已满

bit 3 **TMR4IF**：TMR4 与 PR4 匹配中断标志位
1 = TMR4 与 PR4 发生匹配（必须用软件清零）
0 = TMR4 与 PR4 未发生匹配

bit 2 **CCP5IF**：CCP5 中断标志位
<u>捕捉模式：</u>
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
<u>比较模式：</u>
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
<u>PWM 模式：</u>
在此模式下未使用。

bit 1 **CCP4IF**：CCP4 中断标志位
<u>捕捉模式：</u>
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
<u>比较模式：</u>
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
<u>PWM 模式：</u>
在此模式下未使用。

bit 0 **CCP3IF**：ECCP3 中断标志位
<u>捕捉模式：</u>
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
<u>比较模式：</u>
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
<u>PWM 模式：</u>
在此模式下未使用。

注 **1：** 仅在 100 引脚器件上实现。
2： 仅在 80 引脚和 100 引脚器件上实现。

9.3 PIE 寄存器

PIE 寄存器包含各外设中断的允许位。根据外设中断源的数量，有三个外设中断允许寄存器 （PIE1、 PIE2 和 PIE3）。当 IPEN = 0 时，要允许任一外设中断，必须将 PEIE 位置 1。

寄存器 9-7： PIE1：外设中断允许寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE[(1)]	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **PSPIE：**并行从动端口读 / 写中断允许位 [(1)]
1 = 允许
0 = 禁止

bit 6 **ADIE：**A/D 转换器中断允许位
1 = 允许
0 = 禁止

bit 5 **RC1IE：**EUSART1 接收中断允许位
1 = 允许
0 = 禁止

bit 4 **TX1IE：**EUSART1 发送中断允许位
1 = 允许
0 = 禁止

bit 3 **SSP1IE：**主控同步串口 1 中断允许位
1 = 允许
0 = 禁止

bit 2 **CCP1IE：**ECCP1 中断允许位
1 = 允许
0 = 禁止

bit 1 **TMR2IE：**TMR2 与 PR2 匹配中断允许位
1 = 允许
0 = 禁止

bit 0 **TMR1IE：**TMR1 溢出中断允许位
1 = 允许
0 = 禁止

注 **1：** 仅在单片机模式下的 100 引脚器件上实现。

寄存器 9-8： PIE2：外设中断允许寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **OSCFIE**：振荡器失效中断允许位
1 = 允许
0 = 禁止

bit 6 **CMIE**：比较器中断允许位
1 = 允许
0 = 禁止

bit 5 **ETHIE**：以太网模块中断允许位
1 = 允许
0 = 禁止

bit 4 **保留**：保持为 0

bit 3 **BCL1IE**：总线冲突中断允许位 （MSSP1 模块）
1 = 允许
0 = 禁止

bit 2 **未用**：读为 0

bit 1 **TMR3IE**：TMR3 溢出中断允许位
1 = 允许
0 = 禁止

bit 0 **CCP2IE**：ECCP2 中断允许位
1 = 允许
0 = 禁止

寄存器 9-9： PIE3：外设中断允许寄存器 3

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IE[1]	BCL2IE[1]	RC2IE[2]	TX2IE[2]	TMR4IE	CCP5IE	CCP4IE	CCP3IE
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SSP2IE**：主控同步串口 2 中断允许位 [1]
1 = 允许
0 = 禁止

bit 6 **BCL2IE**：总线冲突中断允许位 （MSSP2 模块） [1]
1 = 允许
0 = 禁止

bit 5 **RC2IE**：EUSART2 接收中断允许位 [2]
1 = 允许
0 = 禁止

bit 4 **TX2IE**：EUSART2 发送中断允许位 [2]
1 = 允许
0 = 禁止

bit 3 **TMR4IE**：TMR4 与 PR4 匹配中断允许位
1 = 允许
0 = 禁止

bit 2 **CCP5IE**：CCP5 中断允许位
1 = 允许
0 = 禁止

bit 1 **CCP4IE**：CCP4 中断允许位
1 = 允许
0 = 禁止

bit 0 **CCP3IE**：ECCP3 中断允许位
1 = 允许
0 = 禁止

注 **1**： 仅在 100 引脚器件上实现。
2： 仅在 80 引脚和 100 引脚器件上实现。

9.4 IPR 寄存器

IPR 寄存器包含各外设中断的优先级位。根据外设中断源的数量，有三个外设中断优先级寄存器（IPR1、IPR2 和 IPR3）。使用优先级位时，要求将中断优先级允许（IPEN）位置 1。

寄存器 9-10： IPR1：外设中断优先级寄存器 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
PSPIP[(1)]	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **PSPIP**：并行从动端口读 / 写中断优先级位 [(1)]
1 = 高优先级
0 = 低优先级

bit 6 **ADIP**：A/D 转换器中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **RC1IP**：EUSART1 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **TX1IP**：EUSART1 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **SSP1IP**：主控同步串口 1 中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **CCP1IP**：ECCP1 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR2IP**：TMR2 与 PR2 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR1IP**：TMR1 溢出中断优先级位
1 = 高优先级
0 = 低优先级

注 1： 仅在单片机模式下的 100 引脚器件上实现。

寄存器 9-11： IPR2：外设中断优先级寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	R/W-1
OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **OSCFIP**：振荡器失效中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **CMIP**：比较器中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **ETHIP**：以太网模块中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **保留**：保持为 1

bit 3 **BCL1IP**：总线冲突中断优先级位 （MSSP1 模块）
1 = 高优先级
0 = 低优先级

bit 2 **未用**：读为 0

bit 1 **TMR3IP**：TMR3 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **CCP2IP**：ECCP2 中断优先级位
1 = 高优先级
0 = 低优先级

寄存器 9-12： IPR3：外设中断优先级寄存器 3

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SSP2IP(1)	BCL2IP(1)	RC2IP(2)	TX2IP(2)	TMR4IP	CCP5IP	CCP4IP	CCP3IP
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SSP2IP**：主控同步串口 2 中断优先级位 (1)
1 = 高优先级
0 = 低优先级

bit 6 **BCL2IP**：总线冲突中断优先级位 （MSSP2 模块） (1)
1 = 高优先级
0 = 低优先级

bit 5 **RC2IP**：EUSART2 接收中断优先级位 (2)
1 = 高优先级
0 = 低优先级

bit 4 **TX2IP**：EUSART2 发送中断优先级位 (2)
1 = 高优先级
0 = 低优先级

bit 3 **TMR4IE**：TMR4 与 PR4 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **CCP5IP**：CCP5 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **CCP4IP**：CCP4 中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **CCP3IP**：ECCP3 中断优先级位
1 = 高优先级
0 = 低优先级

注 **1**： 仅在 100 引脚器件上实现。
2： 仅在 80 引脚和 100 引脚器件上实现。

9.5 RCON 寄存器

RCON寄存器中包含的位可用来确定器件上次复位或从空闲或休眠模式唤醒的原因。 RCON 还包含一个可允许中断优先级的位 （IPEN）。

寄存器 9-13： RCON：复位控制寄存器

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7	**IPEN：** 中断优先级使能位 1 = 使能中断优先级 0 = 禁止中断优先级 （PIC16CXXX 兼容模式）
bit 6-5	未用：读为 0
bit 4	**$\overline{RI}$：** RESET 指令标志位 位操作的详细信息，请参见寄存器 4-1。
bit 3	**$\overline{TO}$：** 看门狗定时器超时标志位 位操作的详细信息，请参见寄存器 4-1。
bit 2	**$\overline{PD}$：** 掉电检测标志位 位操作的详细信息，请参见寄存器 4-1。
bit 1	**$\overline{POR}$：** 上电复位状态位 [(2)] 位操作的详细信息，请参见寄存器 4-1。
bit 0	**$\overline{BOR}$：** 欠压复位状态位 位操作的详细信息，请参见寄存器 4-1。

9.6 INTx 引脚中断

RB0/INT0/FLT0、RB1/INT1、RB2/INT2 和 RB3/INT3 引脚上的外部中断都是边沿触发的。如果 INTCON2 寄存器中相应的 INTEDGx 位被置 1 （= 1），则为上升沿触发；如果该位被清零，则为下降沿触发。当 RBx/INTx 引脚上出现一个有效边沿时，相应的标志位 INTxIF 被置 1。通过清零相应的允许位 INTxIE，可禁止该中断。在重新允许该中断前，必须在中断服务程序中先用软件将中断标志位 INTxIF 清零。

如果 INTxIE 位在进入功耗管理模式前被置 1，则所有的外部中断（INT0、INT1、INT2 和 INT3）均能将处理器从功耗管理模式唤醒。如果全局中断允许位 GIE 被置 1，则处理器将在被唤醒之后转移到中断向量处执行程序。

INT1、 INT2 和 INT3 的中断优先级由中断优先级位 INT1IP（INTCON3<6>）、INT2IP（INTCON3<7>）和 INT3IP （INTCON2<1>）中的值决定。没有与 INT0 相关的优先级位。 INT0 始终是一个高优先级的中断源。

9.7 TMR0 中断

在 8 位模式（默认模式）下， TMR0 寄存器的溢出（FFh → 00h）会使 TMR0IF 标志位置 1。在 16 位模式下，TMR0H:TMR0L 寄存器对的溢出（FFFFh → 0000h）会使 TMR0IF 标志位置 1。通过将允许位 TMR0IE（INTCON<5>）置 1 或清零，可以允许或禁止该中断。Timer0 的中断优先级由中断优先级位 TMR0IP（INTCON2<2>）中的值决定。欲进一步了解 Timer0 模块的详细信息，请参见**第 11.0 节 “Timer0 模块”**。

9.8 PORTB 电平变化中断

PORTB<7:4> 上的输入电平变化会将标志位 RBIF（INTCON<0>）置 1。通过将允许位 RBIE（INTCON<3>）置 1 或清零，可以允许或禁止该中断。 PORTB 电平变化中断的优先级由中断优先级位 RBIP（INTCON2<0>）中的值决定。

9.9 中断的现场保护

在中断期间， PC 的返回地址被保存在堆栈中。另外，WREG、 STATUS 和 BSR 寄存器的值被压入快速返回堆栈。如果未使用从中断快速返回功能（见**第 5.3 节 “数据存储器构成”**），那么用户可能需要在进入中断服务程序前，保存 WREG、STATUS 和 BSR 寄存器的值。根据用户的具体应用，还可能需要保存其他寄存器的值。例 9-1 在执行中断服务程序期间，保存并恢复 WREG、 STATUS 和 BSR 寄存器的值。

例 9-1： 将 STATUS、 WREG 和 BSR 寄存器的值保存在 RAM 中

```
MOVWF    W_TEMP                  ; W_TEMP is in virtual bank
MOVFF    STATUS, STATUS_TEMP     ; STATUS_TEMP located anywhere
MOVFF    BSR, BSR_TEMP           ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR           ; Restore BSR
MOVF     W_TEMP, W               ; Restore WREG
MOVFF    STATUS_TEMP, STATUS     ; Restore STATUS
```


10.0 I/O 端口

根据选定的器件和使能的功能，最多有 9 个端口可供使用。 I/O 端口的一些引脚与器件上外设功能复用。一般来说，当外设被使能时，其对应的引脚就不能被用作通用 I/O 引脚。

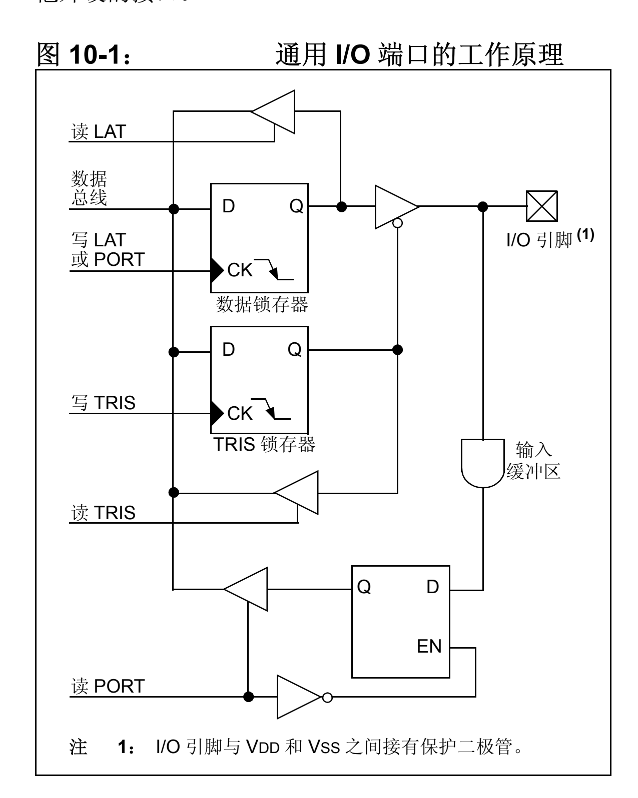
每个端口都有三个工作寄存器。这些寄存器是：

- TRIS 寄存器 （数据方向寄存器）
- PORT 寄存器 （读取器件引脚的电平）
- LAT 寄存器 （输出锁存器）

在对 I/O 引脚驱动值进行读－修改－写时会用到数据锁存器 （LAT 寄存器）。

图 10-1 给出了通用 I/O 端口的简化模型，没有给出到其他外设的接口。

图 10-1： 通用 I/O 端口的工作原理



注 1： I/O 引脚与 VDD 和 VSS 之间接有保护二极管。

10.1 I/O 端口引脚功能

在开发应用程序时，必须考虑到端口引脚的功能。某些引脚上的输出驱动能力比其他引脚要高。同样，某些引脚可以承受高于 VDD 的输入电平。

10.1.1 引脚输出驱动电平

各引脚组的输出引脚驱动能力是不同的，这样可以满足不同应用的需求。 PORTB 和 PORTC 是为驱动较高的负载 （例如 LED）而设计的。外部存储器接口端口（PORTD、PORTE 和 PORTJ）是为驱动中等的负载而设计的。除此之外，所有的其他端口都是为驱动较小的负载而设计的，通常只用于驱动指示器。表 10-1 汇总了输出功能。更多详细信息，请参见**第 27.0 节 “电气特性”**。

表 10-1： 输出驱动电平

端口	驱动	说明
PORTA(1)	低	用于驱动指示器。
PORTF(2)		
PORTG(2)		
PORTH(3)		
PORTD(2)	中	足够高的驱动电平，用于外部存储器接口以及指示器。
PORTE		
PORTJ(3)		
PORTB	高	适合直接输出 LED 驱动电平。
PORTC		

注 1： RA0:RA1 是例外，它们可以直接驱动 LED。
2： 在 64 引脚和 80 引脚器件上部分实现；在 100 引脚器件上完全实现。
3： 在 64 引脚器件上未实现。

10.1.2 输入引脚和电压注意事项

用作器件输入的引脚的电压容差取决于引脚的输入功能。仅用作数字输入的引脚能够承受高达 5.5V 的直流电压，这个电压值是数字逻辑电路的典型电压值。相反，具有模拟输入功能的引脚只能承受最高为 V_{DD} 的电压值。应避免在这些引脚上施加超过 V_{DD} 的电压。表 10-2 汇总了所有引脚的输入能力。更多详细信息，请参见**第 27.0 节 “电气特性”**。

表 10-2： 输入电平

端口或引脚	可承受的最高输入电平	说明
PORTA<5:0>	V_{DD}	只能承受 V_{DD} 以下的输入电压。
PORTF<6:1>(1)		
PORTH<7:4>(2)		
PORTB<7:0>	5.5V	可承受高于 V_{DD} 的输入电压，可用于大部分标准逻辑电路。
PORTC<7:0>		
PORTD<7:0>(1)		
PORTE<7:0>		
PORTF<7>		
PORTG<7:0>(1)		
PORTH<3:0>(2)		
PORTJ<7:0>(2)		

注 1： 在 64 引脚和 80 引脚器件上部分实现；在 100 引脚器件上完全实现。

2： 在 64 引脚器件上不可用。

10.2 PORTA、 TRISA 和 LATA 寄存器

PORTA 是一个 6 位宽的双向端口；它在所有器件上都是完全实现的。对应的数据方向寄存器是 TRISA。将 TRISA 某位置 1 （= 1）时，会将 PORTA 的相应引脚设为输入 （即，使相应的输出驱动器呈高阻状态）。将 TRISA 某位清零 （= 0）时，会将 PORTA 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。

读 PORTA 寄存器将读出相应引脚的状态，而对其进行写操作则是将数据写入端口锁存器。

数据锁存器 （LATA）也是存储器映射的。对 LATA 寄存器执行读－修改－写操作将读写PORTA的输出锁存值。

RA4 引脚与 Timer0 模块的时钟输入引脚复用，成为 RA4/T0CKI 引脚。其他 PORTA 引脚与模拟 $V_{REF}+$ 和 $V_{REF}-$ 输入引脚复用。通过将 ADCON1 寄存器中的 PCFG3:PCFG0 控制位清零或置 1，可将 RA5:RA0 引脚选作 A/D 转换器输入引脚。

> **注：** RA5 和 RA3:RA0 在任何复位时被配置为模拟输入引脚并读为 0。 RA4 被配置为数字输入引脚。

RA4/T0CKI 引脚是施密特触发器输入引脚。所有其他 PORTA引脚都是TTL电平输入和全CMOS驱动输出的。

TRISA 寄存器控制着 PORTA 引脚的方向，即使它们被用作模拟输入。当引脚用于模拟输入时，用户必须确保 TRISA 寄存器中相应的位保持置 1。

RA0 和 RA1 引脚也可以配置为 2 个以太网 LED 指示器的输出。这样配置时，这两个引脚是 PORTA 上仅有的能够产生高输出驱动电平的引脚。

尽管该端口只有 6 位宽，实际上还有一个位 PORTA<7> 是实现为 PORTJ 的弱上拉控制位 RJPU 的。以类似的方式，LATA<7:6> 不是实现为锁存位，而是作为 PORTD 和 PORTE 的上拉控制位 RDPU 和 REPU。将这些位置 1 将使能相应端口的上拉电路。由于其端口引脚不使用， TRISA<7:6> 位未实现。

例 10-1： 初始化 **PORTA**

```
CLRF   PORTA  ;  Initialize PORTA by
              ; clearing output
              ; data latches
CLRF   LATA   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  07h    ; Configure A/D
MOVWF  ADCON1 ; for digital inputs
MOVLW  07h    ; Configure comparators
MOVWF  CMCON  ; for digital input
MOVLW  0CFh   ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISA  ;  Set RA<3:0> as inputs
              ;  RA<5:4> as outputs
```


表 **10-3**： **PORTA** 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RA0/LEDA/AN0	RA0	0	O	DIG	LATA<0> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<0> 数据输入；当使能模拟输入时被禁止。
	LEDA	0	O	DIG	以太网 LEDA 输出；优先于数字数据。
	AN0	1	I	ANA	A/D 输入通道 0。 POR 时的默认输入配置；不影响数字输出。
RA1/LEDB/AN1	RA1	0	O	DIG	LATA<1> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<1> 数据输入；当使能模拟输入时被禁止。
	LEDB	0	O	DIG	以太网 LEDB 输出；优先于数字数据。
	AN1	1	I	ANA	A/D 输入通道 1。 POR 时的默认输入配置；不影响数字输出。
RA2/AN2/VREF-	RA2	0	O	DIG	LATA<2> 数据输出；不受模拟输入影响。当使能 CVREF 输出时被禁止。
		1	I	TTL	PORTA<2> 数据输入。当使能模拟功能时被禁止；当使能 CVREF 输出时被禁止。
	AN2	1	I	ANA	A/D 输入通道 2 和比较器 C2+ 输入。POR 时的默认输入配置；不受模拟输出影响。
	VREF-	1	I	ANA	A/D 和比较器低参考电压输入。
RA3/AN3/VREF+	RA3	0	O	DIG	LATA<3> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<3> 数据输入；当使能模拟输入时被禁止。
	AN3	1	I	ANA	A/D 输入通道 3。 POR 时的默认输入配置。
	VREF+	1	I	ANA	A/D 高参考电压输入。
RA4/T0CKI	RA4	0	O	DIG	LATA<4> 数据输出。
		1	I	ST	PORTA<4> 数据输入； POR 时的默认配置。
	T0CKI	x	I	ST	Timer0 的时钟输入。
RA5/AN4	RA5	0	O	DIG	LATA<5> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<5> 数据输入；当使能模拟输入时被禁止。
	AN4	1	I	ANA	A/D 输入通道 4。 POR 时的默认配置。

图注： O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲区输入，TTL = TTL 缓冲区输入，x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

表 **10-4**： 与 **PORTA** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTA	RJPU(1)	—	RA5	RA4	RA3	RA2	RA1	RA0	62
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	62
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	61
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	60

图注： — = 未用，读为 0。 PORTA 不使用阴影单元。

注 1： 仅在 80 引脚和 100 引脚器件中实现。

10.3 PORTB、 TRISB 和 LATB 寄存器

PORTB 是一个 8 位宽的双向端口；它在所有器件上都是完全实现的。对应的数据方向寄存器是 TRISB。将 TRISB 某位置 1 （= 1）时，会将 PORTB 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISB 某位清零 （= 0）时，会将 PORTB 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。PORTB 上的所有引脚都是仅数字引脚，并且可以承受高达 5.5V 的电压。

数据锁存器 （LATB）也是存储器映射的。对 LATB 寄存器执行读－修改－写操作将读写 PORTB 的输出锁存值。

例 10-2： 初始化 PORTB

```
CLRF   PORTB  ; Initialize PORTB by
              ; clearing output
              ; data latches
CLRF   LATB   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0CFh   ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISB  ; Set RB<3:0> as inputs
              ; RB<5:4> as outputs
              ; RB<7:6> as inputs
```

每个 PORTB 引脚都具有内部弱上拉电路。一个控制位即可接通所有上拉电路。这是通过清零 $\overline{\text{RBPU}}$ 位（INTCON2<7>）实现的。当端口引脚被配置为输出时，其弱上拉电路会自动切断。上电复位会禁止弱上拉电路。

PORTB 的四个引脚 （RB7:RB4）具有电平变化中断功能。仅当将这些引脚配置为输入时，才可使用此中断功能（即当 RB7:RB4 中的任何一个引脚被配置为输出时，该引脚将不再具有电平变化中断功能） 。将输入引脚（RB7:RB4）上的电平与 PORTB 上次读入锁存器的旧值进行比较。对 RB7:RB4 上的 “不匹配”输出进行或运算，产生 RB 端口电平变化中断，并将标志位 RBIF（INTCON<0>）置 1。

该中断可唤醒处于功耗管理模式下的器件。用户可用以下方式在中断服务程序中清除该中断：

a） 读或写 PORTB（`MOVFF (ANY), PORTB` 指令除外）。这将结束不匹配条件。

b） 将标志位 RBIF 清零。

不匹配条件将继续把标志位 RBIF 置 1。读 PORTB 将结束不匹配条件并允许将标志位 RBIF 清零。

建议使用电平变化中断功能实现按键唤醒操作，以及那些仅用到 PORTB 的电平变化中断功能的操作。在使用电平变化中断功能时，建议不要查询 PORTB 的状态。

对工作于扩展单片机模式下的 100 引脚器件，可通过将 CCP2MX 配置位清零，将 RB3 配置为 ECCP2 模块和增强型 PWM 输出 P2A 的备用外设引脚。如果器件处于单片机模式下， ECCP2 的替代引脚为 RE7。至于其他 ECCP2 配置，用户必须确保为要进行的操作将 TRISB<3> 位进行相应的设置。

 超前信息

表 **10-5**： **PORTB** 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RB0/INT0/FLT0	RB0	0	O	DIG	LATB<0> 数据输出。
		1	I	TTL	PORTB<0> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	INT0	1	I	ST	外部中断 0 输入。
	FLT0	1	I	ST	增强型 PWM 故障输入 （ECCP1 模块）；通过软件使能。
RB1/INT1	RB1	0	O	DIG	LATB<1> 数据输出。
		1	I	TTL	PORTB<1> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	INT1	1	I	ST	外部中断 1 输入。
RB2/INT2	RB2	0	O	DIG	LATB<2> 数据输出。
		1	I	TTL	PORTB<2> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	INT2	1	I	ST	外部中断 2 输入。
RB3/INT3/ECCP2/P2A	RB3	0	O	DIG	LATB<3> 数据输出。
		1	I	TTL	PORTB<3> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	INT3	1	I	ST	外部中断 3 输入。
	ECCP2[(1)]	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先于端口数据。
		1	I	ST	CCP2 捕捉输入。
	P2A[(1)]	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A。可以在增强型 PWM 关闭期间被配置为三态。优先于端口数据。
RB4/KBI0	RB4	0	O	DIG	LATB<4> 数据输出。
		1	I	TTL	PORTB<4> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	KBI0	1	I	TTL	引脚电平变化中断。
RB5/KBI1	RB5	0	O	DIG	LATB<5> 数据输出。
		1	I	TTL	PORTB<5> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	KBI1	1	I	TTL	引脚电平变化中断。
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> 数据输出。
		1	I	TTL	PORTB<6> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	KBI2	1	I	TTL	引脚电平变化中断。
	PGC	x	I	ST	供 ICSP 和 ICD 工作使用的串行执行 （ICSP™ ）时钟输入。[(2)]
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> 数据输出。
		1	I	TTL	PORTB<7> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉电路。
	KBI3	1	I	TTL	引脚电平变化中断。
	PGD	x	O	DIG	供 ICSP 和 ICD 工作使用的串行执行数据输出。[(2)]
		x	I	ST	供 ICSP 和 ICD 工作使用的串行执行数据输入。[(2)]

图注： O = 输出， I = 输入， DIG = 数字输出， ST = 施密特缓冲区输入， TTL = TTL 缓冲区输入， x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 **1**： 当 CCP2MX 配置位被清零时， ECCP2/P2A 的备用配置 （扩展单片机模式下的 100 引脚器件）。默认分配是 RC1。

2： 当使能 ICSP 或 ICD 时，禁止所有其他引脚功能。

表 10-6： 与 PORTB 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	62
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	62
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	61
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	59
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	59

图注： PORTB 不使用阴影单元。

10.4 PORTC、 TRISC 和 LATC 寄存器

PORTC 是一个 8 位宽的双向端口；它在所有器件上都是完全实现的。对应的数据方向寄存器是 TRISC。将 TRISC 某位置 1 （= 1）时，会将 PORTC 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISC 某位清零 （= 0）时，会将 PORTC 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。PORTC 引脚 RC2 到 RC7 只能作为数字引脚并且可以承受高达 5.5V 的输入电压。

数据锁存器 （LATC）也是存储器映射的。对 LATC 寄存器执行读－修改－写操作将读写 PORTC 的输出锁存值。

PORTC 与几种外设功能复用 （表 10-7）。这些引脚配有施密特触发输入缓冲区。通常由配置位 CCP2MX 将 RC1配置为ECCP2模块的默认外设引脚和增强型PWM 输出 P2A 引脚 （默认状态， CCP2MX = 1）。

当使能外设功能时，应小心定义每个 PORTC 引脚的 TRIS 位。有些外设会无视 TRIS 位的设置，将引脚定义为输出引脚或输入引脚。用户应该查阅相应的外设章节来正确设置TRIS 位。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

外设对引脚的改写会影响TRISC 寄存器的内容。尽管如此，读 TRISC 总是会返回其当前的内容。

例 10-3： 初始化 PORTC

```
CLRF   PORTC  ; Initialize PORTC by
              ; clearing output
              ; data latches
CLRF   LATC   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0CFh   ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISC  ; Set RC<3:0> as inputs
              ; RC<5:4> as outputs
              ; RC<7:6> as inputs
```

表 10-7： PORTC 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RC0/T1OSO/ T13CKI	RC0	0	O	DIG	LATC<0> 数据输出。
		1	I	ST	PORTC<0> 数据输入。
	T1OSO	x	O	ANA	Timer1 振荡器输出；当使能 Timer1 振荡器时被使能。禁止数字 I/O。
	T13CKI	1	I	ST	Timer1/Timer3 计数器输入。
RC1/T1OSI/ ECCP2/P2A	RC1	0	O	DIG	LATC<1> 数据输出。
		1	I	ST	PORTC<1> 数据输入。
	T1OSI	x	I	ANA	Timer1 振荡器输入；当使能 Timer1 振荡器时被使能。禁止数字 I/O。
	ECCP2(1)	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先于端口数据。
		1	I	ST	CCP2 捕捉输入。
	P2A(1)	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A。可以在增强型 PWM 关闭期间被配置为三态。优先于端口数据。
RC2/ECCP1/ P1A	RC2	0	O	DIG	LATC<2> 数据输出。
		1	I	ST	PORTC<2> 数据输入。
	ECCP1	0	O	DIG	CCP1 比较输出和 CCP1 PWM 输出；优先于端口数据。
		1	I	ST	CCP1 捕捉输入。
	P1A	0	O	DIG	ECCP1 增强型 PWM 输出，通道 A。可以在增强型 PWM 关闭期间被配置为三态。优先于端口数据。
RC3/SCK1/ SCL1	RC3	0	O	DIG	LATC<3> 数据输出。
		1	I	ST	PORTC<3> 数据输入。
	SCK1	0	O	DIG	SPI 时钟输出 （MSSP1 模块）；优先于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP1 模块）。
	SCL1	0	O	DIG	I^{2}C™ 时钟输出 （MSSP1 模块）；优先于端口数据。
		1	I	ST	I^{2}C 时钟输入 （MSSP1 模块）；输入类型取决于模块设置。
RC4/SDI1/ SDA1	RC4	0	O	DIG	LATC<4> 数据输出。
		1	I	ST	PORTC<4> 数据输入。
	SDI1	1	I	ST	SPI 数据输入 （MSSP1 模块）。
	SDA1	1	O	DIG	I^{2}C 数据输出 （MSSP1 模块）；优先于端口数据。
		1	I	ST	I^{2}C 数据输入 （MSSP1 模块）；输入类型取决于模块设置。
RC5/SDO1	RC5	0	O	DIG	LATC<5> 数据输出。
		1	I	ST	PORTC<5> 数据输入。
	SDO1	0	O	DIG	SPI 数据输出 （MSSP1 模块）；优先于端口数据。
RC6/TX1/CK1	RC6	0	O	DIG	LATC<6> 数据输出。
		1	I	ST	PORTC<6> 数据输入。
	TX1	1	O	DIG	同步串行数据输出 （EUSART1 模块）；优先于端口数据。
	CK1	1	O	DIG	同步串行数据输入 （EUSART1 模块）。用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART1 模块）。
RC7/RX1/DT1	RC7	0	O	DIG	LATC<7> 数据输出。
		1	I	ST	PORTC<7> 数据输入。
	RX1	1	I	ST	异步串行接收数据输入 （EUSART1 模块）。
	DT1	1	O	DIG	同步串行数据输出 （EUSART1 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （EUSART1 模块）。用户必须将其配置为输入。

图注： O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲区输入，x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 1： 当配置位 CCP2MX 置 1 时，对 ECCP2/P2A 进行默认分配。

 超前信息

表 **10-8**：　与 **PORTC** 相关的寄存器汇总

名称	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	复位值所在页
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	62
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	62
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	61

10.5 PORTD、 TRISD 和 LATD 寄存器

PORTD 以两种方式实现为双向端口：

- 64 引脚和 80 引脚器件：3 位宽 （RD<2:0>）
- 100 引脚器件：8 位宽 （RD<7:0>）

对应的数据方向寄存器是 TRISD。将 TRISD 某位置 1（= 1）时，会将 PORTD 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISD 某位清零（= 0）时，会将 PORTD 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。PORTD上的所有引脚都是仅数字引脚，并且可以承受高达 5.5V 的电压。

数据锁存器 （LATD）也是存储器映射的。对 LATD 寄存器执行读－修改－写操作将读写 PORTD 的输出锁存值。

PORTD 上的所有引脚都配有施密特触发输入缓冲区。每个引脚都可被单独配置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

在 100 引脚器件上，PORTD 与系统总线复用作为外部存储器接口的一部分。 I/O 端口和其他功能只有在通过将 EBDIS 位（MEMCON<7>）置 1 而禁止接口功能时才可用。当使能接口时， PORTD 是复用的地址 / 数据总线的低字节 （AD7:AD0）。 TRISD 位也被改写。

PORTD 的每个引脚都具有内部弱上拉电路。提供上拉电路是为了保证上电时外部存储器接口输入为一个已知状态。单个控制位可以关闭所有上拉电路。可以通过清零 RDPU 位 （LATA<7>）来实现。当端口引脚被配置为输出时，其弱上拉电路会自动切断。在所有器件复位时上拉电路被使能。

在 100 引脚器件上，还可通过将控制位 PSPMODE（PSPCON<4>）置 1，将 PORTD 配置为 8 位宽的并行单片机端口。在此模式下，并行端口数据优先于其他数字 I/O（但外部存储器接口除外）数据。当激活并行端口时，输入缓冲区为 TTL。更多信息，请参见**第 10.11 节“并行从动端口”**。

例 10-4： 初始化 PORTD

```
CLRF   PORTD  ;  Initialize PORTD by
              ; clearing output
              ; data latches
CLRF   LATD   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0CFh   ;  Value used to
              ; initialize data
              ; direction
MOVWF  TRISD  ;  Set RD<3:0> as inputs
              ;  RD<5:4> as outputs
              ;  RD<7:6> as inputs
```

 超前信息

表 **10-9**： **PORTD** 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD0/AD0/PSP0（RD0/P1B）	RD0	0	O	DIG	LATD<0> 数据输出。
		1	I	ST	PORTD<0> 数据输入。
	AD0(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 0 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 0 输入。(2)
	PSP0(1)	x	O	DIG	PSP 读输出数据 （LATD<0>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	P1B(3)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B ；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RD1/AD1/PSP1（RD1/ECCP3/P3A）	RD1	0	O	DIG	LATD<1> 数据输出。
		1	I	ST	PORTD<1> 数据输入。
	AD1(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 1 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 1 输入。(2)
	PSP1(1)	x	O	DIG	PSP 读输出数据 （LATD<1>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	ECCP3(3)	0	O	DIG	ECCP3 比较和 PWM 输出；优先于端口数据。
		1	I	ST	ECCP3 捕捉输入。
	P3A(3)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 A ；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RD2/AD2/PSP2（RD2/CCP4/P3D）	RD2	0	O	DIG	LATD<2> 数据输出。
		1	I	ST	PORTD<2> 数据输入。
	AD2(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 2 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 2 输入。(2)
	PSP2(1)	x	O	DIG	PSP 读输出数据 （LATD<2>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	CCP4(3)	0	O	DIG	CCP4 比较输出和 CCP4 PWM 输出；优先于端口数据。
		1	I	ST	CCP4 捕捉输入。
	P3D(3)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 D ；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RD3/AD3/PSP3(1)	RD3(1)	0	O	DIG	LATD<3> 数据输出。
		1	I	ST	PORTD<3> 数据输入。
	AD3(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 3 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 3 输入。(2)
	PSP3(1)	x	O	DIG	PSP 读输出数据 （LATD<3>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。

图注： O = 输出， I = 输入， DIG = 数字输出， ST = 施密特缓冲区输入， TTL = TTL 缓冲区输入， x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 **1**： 这些功能或端口引脚仅在 100 引脚器件上实现。

2： 外部存储器接口 I/O 优先于所有其他数字和 PSP I/O。

3： 这些功能仅在 64 引脚器件的该引脚上实现；对于所有其他器件，它们与 RE6/RH7 （P1B）、 RG0 （ECCP3/P3A）或 RG3 （CCP4/P3D）复用。

表 10-9： PORTD 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD4/AD4/PSP4/SDO2(1)	RD4(1)	0	O	DIG	LATD<4> 数据输出。
		1	I	ST	PORTD<4> 数据输入。
	AD4(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 4 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 4 输入。(2)
	PSP4(1)	x	O	DIG	PSP 读输出数据 （LATD<4>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	SDO2(1)	0	O	DIG	SPI 数据输出 （MSSP2 模块）；优先于端口数据。
RD5/AD5/PSP5/SDI2/SDA2(1)	RD5(1)	0	O	DIG	LATD<5> 数据输出。
		1	I	ST	PORTD<5> 数据输入。
	AD5(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 5 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 5 输入。(2)
	PSP5(1)	x	O	DIG	PSP 读输出数据 （LATD<5>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	SDI2(1)	1	I	ST	SPI 数据输入 （MSSP2 模块）。
	SDA2(1)	1	O	DIG	I^2C™ 数据输出 （MSSP2 模块）；优先于端口数据。
		1	I	ST	I^2C 数据输入 （MSSP2 模块）；输入类型取决于模块设置。
RD6/AD6/PSP6/SCK2/SCL2(1)	RD6(1)	0	O	DIG	LATD<6> 数据输出。
		1	I	ST	PORTD<6> 数据输入。
	AD6(1)	x	O	DIG-3	外部存储器接口，地址 / 数据 bit 6 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 6 输入。(1)
	PSP6(1)	x	O	DIG	PSP 读输出数据 （LATD<6>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	SCK2(1)	0	O	DIG	SPI 时钟输出 （MSSP2 模块）；优先于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP2 模块）。
	SCL2(1)	0	O	DIG	I^2C 时钟输出 （MSSP2 模块）；优先于端口数据。
		1	I	ST	I^2C 时钟输入 （MSSP2 模块）；输入类型取决于模块设置。
RD7/AD7/PSP7/$\overline{SS2}$(1)	RD7(1)	0	O	DIG	LATD<7> 数据输出。
		1	I	ST	PORTD<7> 数据输入。
	AD7(1)	x	O	DIG	外部存储器接口，地址 / 数据 bit 7 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 7 输入。(1)
	PSP7(1)	x	O	DIG	PSP 读输出数据 （LATD<7>）；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	$\overline{SS2}$(1)	x	I	TTL	MSSP 的从动选择输入 （MSSP2 模块）。

图注： O = 输出，I = 输入， DIG = 数字输出， ST = 施密特缓冲区输入， TTL = TTL 缓冲区输入， x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 1： 这些功能或端口引脚仅在 100 引脚器件上实现。

2： 外部存储器接口 I/O 优先于所有其他数字和 PSP I/O。

3： 这些功能仅在 64 引脚器件的该引脚上实现；对于所有其他器件，它们与 RE6/RH7 （P1B）、 RG0 （ECCP3/P3A）或 RG3 （CCP4/P3D）复用。

表 **10-10**： 与 **PORTD** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD	RD7[1]	RD6[1]	RD5[1]	RD4[1]	RD3[1]	RD2	RD1	RD0	62
LATD	LATD7[1]	LATD6[1]	LATD5[1]	LATD4[1]	LATD3[1]	LATD2	LATD1	LATD0	62
TRISD	TRISD7[1]	TRISD6[1]	TRISD5[1]	TRISD4[1]	TRISD3[1]	TRISD2	TRISD1	TRISD0	61
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	62

图注： PORTD 不使用阴影单元。

注 **1**： 在 64 引脚和 80 引脚器件上未实现，读为 0。

10.6 PORTE、 TRISE 和 LATE 寄存器

PORTE 以两种不同方式实现为双向端口：

- 64 引脚器件：6 位宽 （RE<5:0>）
- 80 引脚和 100 引脚器件：8 位宽 （RE<7:0>）

对应的数据方向寄存器是 TRISE。将 TRISE 某位置 1（= 1）时，会将 PORTE 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISE 某位清零（= 0）时，会将 PORTE 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。PORTE上的所有引脚都是仅数字引脚，并且可以承受高达 5.5V 的电压。

数据锁存器 （LATE）也是存储器映射的。对 LATE 寄存器执行读－修改－写操作将读写 PORTE 的输出锁存值。

PORTE 上的所有引脚都配有施密特触发输入缓冲区。每个引脚都可被单独配置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

在 100 引脚器件上， PORTE 与系统总线复用作为外部存储器接口的一部分。 I/O 端口和其他功能只有在通过将 EBDIS 位 （MEMCON<7>）置 1 而禁止接口功能时才可用。当使能接口时，PORTE 是复用的地址 / 数据总线的高字节 （AD15:AD8）。 TRISE 位也被改写。

PORTE 的每个引脚都具有内部弱上拉电路。提供上拉电路是为了保证上电时外部存储器接口输入为一个已知状态。单个控制位可以关闭所有上拉电路。可以通过清零 REPU 位 （LATA<6>）来实现。当端口引脚被配置为输出时，其弱上拉电路会自动切断。发生任何器件复位时，上拉功能会被禁止。

PORTE 也可与以下这些功能复用：ECCP1 和 ECCP3 的增强型 PWM 输出 B 和 C，以及 ECCP2 的输出 B、C 和 D。对于 80 引脚和 100 引脚器件，它们的默认分配是在 PORTE<6:0> 上。对于 64 引脚器件，它们的默认分配是在 PORTE<5:0> 和 PORTD<0> 上。在 80 引脚和 100 引脚器件上， ECCP1 和 ECCP3 的输出复用受 ECCPMX 配置位的控制。清零该位将把 P1B/P1C 和 P3B/P3C 输出重新分配到 PORTH。

对于运行于单片机模式的 80 引脚和 100 引脚器件，引脚 RE7 可配置为 ECCP2 模块和增强型 PWM 输出 P2A 的备用外设引脚。这是通过将 CCP2MX 配置位清零实现的。

当 PORTD 上的并行从动端口为激活状态时， 3 个 PORTE 引脚（RE0、RE1 和 RE2）将被配置为该端口的数字控制输入。表 10-11 中汇总了此控制功能。当 PSPMODE 控制位（PSPCON<4>）被置 1 时，会自动重新配置。用户还必须保证将相应的 TRISE 位置 1 以便将这些引脚配置为数字输入。

例 10-5： 初始化 PORTE

```
CLRF   PORTE  ; Initialize PORTE by
              ; clearing output
              ; data latches
CLRF   LATE   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  03h    ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISE  ; Set RE<1:0> as inputs
              ; RE<7:2> as outputs
```


表 10-11： PORTE 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE0/AD8/$\overline{RD}$/P2D	RE0	0	O	DIG	LATE<0> 数据输出。
		1	I	ST	PORTE<0> 数据输入。
	AD8[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 8 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 8 输入。[2]
	$\overline{RD}$[6]	1	I	TTL	并行从动端口读使能控制输入。
	P2D	0	O	DIG	ECCP2 增强型 PWM 输出，通道 D；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RE1/AD9/$\overline{WR}$/P2C	RE1	0	O	DIG	LATE<1> 数据输出。
		1	I	ST	PORTE<1> 数据输入。
	AD9[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 9 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 9 输入。[2]
	$\overline{WR}$[6]	1	I	TTL	并行从动端口写使能控制输入。
	P2C	0	O	DIG	ECCP2 增强型 PWM 输出，通道 C；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RE2/AD10/$\overline{CS}$/P2B	RE2	0	O	DIG	LATE<2> 数据输出。
		1	I	ST	PORTE<2> 数据输入。
	AD10[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 10 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 10 输入。[2]
	$\overline{CS}$[6]	1	I	TTL	并行从动端口片选控制输入。
	P2B	0	O	DIG	ECCP2 增强型 PWM 输出，通道 B；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RE3/AD11/P3C	RE3	0	O	DIG	LATE<3> 数据输出。
		1	I	ST	PORTE<3> 数据输入。
	AD11[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 11 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 11 输入。[2]
	P3C[3]	0	O	DIG	ECCP3 增强型 PWM 输出，通道 C；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RE4/AD12/P3B	RE4	0	O	DIG	LATE<4> 数据输出。
		1	I	ST	PORTE<4> 数据输入。
	AD12[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 12 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 12 输入。[2]
	P3B[3]	0	O	DIG	ECCP3 增强型 PWM 输出，通道 B；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。

图注： O = 输出，I = 输入，DIG = 数字输出，ST = 施密特缓冲区输入，TTL = TTL 缓冲区输入，x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 1： 仅在 100 引脚器件上实现了 EMB 功能。
2： 外部存储器接口 I/O 优先于所有其他数字和 PSP I/O。
3： ECCPMX 配置位置 1 时 P1B/P1C 和 P3B/P3C 的默认分配 （80 引脚和 100 引脚器件）。
4： 在 64 引脚器件上未实现。
5： CCP2MX 配置位清零时 ECCP2/P2A 的备用分配 （单片机模式中的 80 引脚和 100 引脚器件）。
6： 在 64 引脚和 80 引脚器件上实现。

表 10-11：　PORTE 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE5/AD13/ P1C	RE5	0	O	DIG	LATE<5> 数据输出。
		1	I	ST	PORTE<5> 数据输入。
	AD13[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 13 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 13 输入。[2]
	P1C[3]	0	O	DIG	ECCP1 增强型 PWM 输出，通道 C ；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RE6/AD14/ P1B[4]	RE6	0	O	DIG	LATE<6> 数据输出。
		1	I	ST	PORTE<6> 数据输入。
	AD14[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 14 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 14 输入。[2]
	P1B[3]	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B ；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RE7/AD15/ ECCP2/P2A[4]	RE7	0	O	DIG	LATE<7> 数据输出。
		1	I	ST	PORTE<7> 数据输入。
	AD15[1]	x	O	DIG	外部存储器接口，地址 / 数据 bit 15 输出。[2]
		x	I	TTL	外部存储器接口，数据 bit 15 输入。[2]
	ECCP2[5]	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先于端口数据。
		1	I	ST	CCP2 捕捉输入。
	P2A[5]	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A ；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。

图注：　O = 输出， I = 输入， DIG = 数字输出， ST = 施密特缓冲区输入， TTL = TTL 缓冲区输入， x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注　1： 仅在 100 引脚器件上实现了 EMB 功能。
2： 外部存储器接口 I/O 优先于所有其他数字和 PSP I/O。
3： ECCPMX 配置位置 1 时 P1B/P1C 和 P3B/P3C 的默认分配 （80 引脚和 100 引脚器件）。
4： 在 64 引脚器件上未实现。
5： CCP2MX 配置位清零时 ECCP2/P2A 的备用分配 （单片机模式中的 80 引脚和 100 引脚器件）。
6： 在 64 引脚和 80 引脚器件上实现。

表 10-12：　与 PORTE 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTE	RE7[1]	RE6[1]	RE5	RE4	RE3	RE2	RE1	RE0	62
LATE	LATE7[1]	LATE6[1]	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	62
TRISE	TRISE7[1]	TRISE6[1]	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	61
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	62

图注：　— = 未用，读为 0。 PORTE 不使用阴影单元。

注　1： 在 64 引脚器件上未实现，读为 0。

超前信息

10.7 PORTF、 LATF 和 TRISF 寄存器

PORTF 以两种不同方式实现为双向端口：

- 64 引脚和 80 引脚器件：7 位宽 （RF<7:1>）
- 100 引脚器件：8 位宽 （RF<7:0>）

对应的数据方向寄存器是 TRISF。将 TRISF 某位置 1（= 1）时，会将 PORTF 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISF 某位清零（= 0）时，会将 PORTF 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。只有 PORTF 的引脚 7 没有模拟输入；它是唯一可以承受高达 5.5V 的电压的引脚。

数据锁存器（LATF）也是存储器映射的。对 LATF 寄存器执行读－修改－写操作将读写 PORTF 的输出锁存值。

PORTF 上的所有引脚都配有施密特触发输入缓冲区。每个引脚都可被单独配置为输入或输出。

PORTF 与几种模拟外设功能复用，包括 A/D 转换器、比较器输入和比较器输出。可以通过设置 CMCON 寄存器，将 RF1 到 RF6 引脚用作比较器输入或输出。要将 RF6:RF1 用作数字输入，还必须关闭比较器。

注	
1：	当器件复位时，引脚 RF6:RF1 被配置为模拟输入并读为 0。
2：	要将 PORTF 配置为数字 I/O，可以关闭比较器并设置 ADCON1 的值。

例 10-6： 初始化 PORTF

```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   07h     ;
MOVWF   CMCON   ; Turn off comparators
MOVLW   0Fh     ;
MOVWF   ADCON1  ; Set PORTF as digital I/O
MOVLW   0CEh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISF   ; Set RF3:RF1 as inputs
                ; RF5:RF4 as outputs
                ; RF7:RF6 as inputs
```

表 10-13： PORTF 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RF0/AN5[1]	RF0[1]	0	O	DIG	LATF<0> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<0> 数据输入；当使能模拟输入时被禁止。
	AN5[1]	1	I	ANA	A/D 输入通道 5。 POR 时的默认配置。
RF1/AN6/ C2OUT	RF1	0	O	DIG	LATF<1> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<1> 数据输入；当使能模拟输入时被禁止。
	AN6	1	I	ANA	A/D 输入通道 6。 POR 时的默认配置。
	C2OUT	0	O	DIG	比较器 2 的输出；优先于端口数据。
RF2/AN7/ C1OUT	RF2	0	O	DIG	LATF<2> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<2> 数据输入；当使能模拟输入时被禁止。
	AN7	1	I	ANA	A/D 输入通道 7。 POR 时的默认配置。
	C1OUT	0	O	TTL	比较器 1 的输出；优先于端口数据。
RF3/AN8	RF3	0	O	DIG	LATF<3> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<3> 数据输入；当使能模拟输入时被禁止。
	AN8	1	I	ANA	A/D 输入通道 8 和比较器 C2+ 输入。 POR 时的默认输入配置；不受模拟输出影响。
RF4/AN9	RF4	0	O	DIG	LATF<4> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<4> 数据输入；当使能模拟输入时被禁止。
	AN9	1	I	ANA	A/D 输入通道 9 和比较器 C2- 输入。 POR 时的默认输入配置；不影响数字输出。
RF5/AN10/ CVREF	RF5	0	O	DIG	LATF<5> 数据输出；不受模拟输入影响。当使能 CVREF 输出时被禁止。
		1	I	ST	PORTF<5> 数据输入；当使能模拟输入时被禁止。当使能 CVREF 输出时被禁止。
	AN10	1	I	ANA	A/D 输入通道 10 和比较器 C1+ 输入。 POR 时的默认输入配置。
	CVREF	x	O	ANA	比较器参考电压输出。使能该功能将禁止数字 I/O。
RF6/AN11	RF6	0	O	DIG	LATF<6> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<6> 数据输入；当使能模拟输入时被禁止。
	AN11	1	I	ANA	A/D 输入通道 11 和比较器 C1- 输入。 POR 时的默认输入配置；不影响数字输出。
RF7/$\overline{SS1}$	RF7	0	O	DIG	LATF<7> 数据输出。
		1	I	ST	PORTF<7> 数据输入。
	$\overline{SS1}$	1	I	TTL	MSSP 的从动选择输入 （MSSP1 模块）。

图注： O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲区输入，TTL = TTL 缓冲区输入，x = 无关位 （TRIS 位不影响端口方向或在此可忽略）。

注 1： 仅在 100 引脚器件上实现。

表 10-14： 与 PORTF 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0[1]	62
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0[1]	62
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0[1]	61
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	60
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	60
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	60

图注： — = 未用，读为 0。 PORTF 不使用阴影单元。

注 1： 仅在 100 引脚器件上实现。

超前信息

10.8 PORTG、 TRISG 和 LATG 寄存器

取决于特定的器件， PORTG 以三种方式之一实现为双向端口：

- 64 引脚器件：1 位宽 （RG<4>）
- 80 引脚器件：5 位宽 （RG<4:0>）
- 100 引脚器件：8 位宽 （RG<7:0>）

对应的数据方向寄存器是 TRISG。将 TRISG 某位置 1（= 1）时，会将 PORTG 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISG 某位清零（= 0）时，会将 PORTG 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。 PORTG 上的所有引脚都是仅数字引脚，并且可以承受高达 5.5V 的电压。

数据锁存器 （LATG）也是存储器映射的。对 LATG 寄存器执行读－修改－写操作将读写 PORTG 的输出锁存值。

PORTG 与 80 引脚及 100 引脚器件上的 EUSART2 功能复用 （表 10-15）。 PORTG 引脚配有施密特触发输入缓冲区。

当使能外设功能时，应小心定义每个 PORTG 引脚的 TRIS 位。有些外设会无视 TRIS 位的设置，将引脚定义为输出引脚或输入引脚。用户应该查阅相应的外设章节来正确设置 TRIS 位。引脚改写值未装入 TRIS 寄存器中。这将允许对 TRIS 寄存器执行读－修改－写操作而无需担心外设的改写。

例 10-7： 初始化 PORTG

```
CLRF   PORTG  ;  Initialize PORTG by
              ; clearing output
              ; data latches
CLRF   LATG   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  04h    ;  Value used to
              ; initialize data
              ; direction
MOVWF  TRISG  ;  Set RG1:RG0 as outputs
              ; RG2 as input
              ;  RG4:RG3 as inputs
```

表 10-15： **PORTG 功能**

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RG0/ECCP3/ P3A[1]	RG0[1]	0	O	DIG	LATG<0> 数据输出。
		1	I	ST	PORTG<0> 数据输入。
	ECCP3[1]	0	O	DIG	CCP3 比较输出和 PWM 输出；优先于端口数据。
		1	I	ST	CCP3 捕捉输入。
	P3A[1]	0	O	DIG	ECCP3 增强型 PWM 输出，通道 A ；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RG1/TX2/ CK2[1]	RG1[1]	0	O	DIG	LATG<1> 数据输出。
		1	I	ST	PORTG<1> 数据输入。
	TX2[1]	1	O	DIG	同步串行数据输出 （EUSART2 模块）；优先于端口数据。
	CK2[1]	1	O	DIG	同步串行数据输入 （EUSART2 模块）。用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART2 模块）。
RG2/RX2/ DT2[1]	RG2[1]	0	O	DIG	LATG<2> 数据输出。
		1	I	ST	PORTG<2> 数据输入。
	RX2[1]	1	I	ST	异步串行接收数据输入 （EUSART2 模块）。
	DT2[1]	1	O	DIG	同步串行数据输出 （EUSART2 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （EUSART2 模块）。用户必须将其配置为输入。
RG3/CCP4/ P3D[1]	RG3[1]	0	O	DIG	LATG<3> 数据输出。
		1	I	ST	PORTG<3> 数据输入。
	CCP4[1]	0	O	DIG	CCP4 比较输出和 CCP4 PWM 输出；优先于端口数据。
		1	I	ST	CCP4 捕捉输入。
	P3D[1]	0	O	DIG	ECCP3 增强型 PWM 输出，通道 D ；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RG4/CCP5/ P1D	RG4	0	O	DIG	LATG<4> 数据输出。
		1	I	ST	PORTG<4> 数据输入。
	CCP5	0	O	DIG	CCP5 比较输出和 CCP5 PWM 输出；优先于端口数据。
		1	I	ST	CCP5 捕捉输入。
	P1D	0	O	DIG	ECCP1 增强型 PWM 输出，通道 D ；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RG5[2]	RG5[2]	0	O	DIG	LATG<0> 数据输出。
		1	I	ST	PORTG<0> 数据输入。
RG6[2]	RG6[2]	0	O	DIG	LATG<0> 数据输出。
		1	I	ST	PORTG<0> 数据输入。
RG7[2]	RG7[2]	0	O	DIG	LATG<0> 数据输出。
		1	I	ST	PORTG<0> 数据输入。

图注： O = 输出，I = 输入，DIG = 数字输出，ST = 施密特缓冲区输入，x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 **1：** 仅在 80 引脚和 100 引脚器件上实现。

2： 仅在 100 引脚器件上实现。

表 **10-16**： 与 **PORTG** 相关的寄存器汇总

名称	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	复位值所在页
PORTG	RG7(1)	RG6(1)	RG5(1)	RG4	RG3(2)	RG2(2)	RG1(2)	RG0(2)	62
LATG	LATG7(1)	LATG6(1)	LATG5(1)	LATG4	LATG3(2)	LATG2(2)	LATG1(2)	LATG0(2)	62
TRISG	TRISG7(1)	TRISG6(1)	TRISG5(1)	TRISG4	TRISG3(2)	TRISG2(2)	TRISG1(2)	TRISE0(2)	61

注 **1**： 仅在 100 引脚器件上实现。

2： 仅在 80 引脚和 100 引脚器件上实现。

10.9 PORTH、 LATH 和 TRISH 寄存器

注：	PORTH 仅在 80 引脚和 100 引脚器件上存在。

PORTH 是一个 8 位宽的双向 I/O 端口；它在 80 引脚和 100 引脚器件上是完全实现的。对应的数据方向寄存器是 TRISH。将 TRISH 某位置 1（= 1）时，会将 PORTH 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISH 某位清零 （= 0）时，会将 PORTH 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。PORTH<3:0> 是仅数字引脚，并且可以承受高达 5.5V 的电压。

数据锁存器 （LATH）也是存储器映射的。对 LATH 寄存器执行读－修改－写操作将读写 PORTH 的输出锁存值。

PORTH 上的所有引脚都配有施密特触发输入缓冲区。每个引脚都可被单独配置为输入或输出。

当使能外部存储器接口时， 4 个 PORTH 引脚被用作该接口的高位地址线。来自接口的地址输出优先于其他数字 I/O。相应的 TRISH 位也被改写。

PORTH 引脚 RH4 到 RH7 与模拟转换器输入复用。通过将 ADCON1 寄存器中的 PCFG3:PCFG0 控制位清零或置 1，可将这些引脚选作模拟输入引脚。

PORTH 也可配置为 ECCP1 和 ECCP3 模块的备用增强型 PWM 输出通道 B 和 C。这是通过将 ECCPMX 配置位清零实现的。

例 10-8： 初始化 PORTH

```
CLRF   PORTH      ; Initialize PORTH by
                  ; clearing output
                  ; data latches
CLRF   LATH       ; Alternate method
                  ; to clear output
                  ; data latches
MOVLW  0Fh        ; Configure PORTH as
MOVWF  ADCON1     ; digital I/O
MOVLW  0CFh       ; Value used to
                  ; initialize data
                  ; direction
MOVWF  TRISH      ; Set RH3:RH0 as inputs
                  ; RH5:RH4 as outputs
                  ; RH7:RH6 as inputs
```

 超前信息

表 10-17： PORTH 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RH0/A16	RH0	0	O	DIG	LATH<0> 数据输出。
		1	I	ST	PORTH<0> 数据输入。
	A16(1)	x	O	DIG	外部存储器接口，地址线 16。优先于端口数据。
RH1/A17	RH1	0	O	DIG	LATH<1> 数据输出。
		1	I	ST	PORTH<1> 数据输入。
	A17(1)	x	O	DIG	外部存储器接口，地址线 17。优先于端口数据。
RH2/A18	RH2	0	O	DIG	LATH<2> 数据输出。
		1	I	ST	PORTH<2> 数据输入。
	A18(1)	x	O	DIG	外部存储器接口，地址线 18。优先于端口数据。
RH3/A19	RH3	0	O	DIG	LATH<3> 数据输出。
		1	I	ST	PORTH<3> 数据输入。
	A19(1)	x	O	DIG	外部存储器接口，地址线 19。优先于端口数据。
RH4/AN12/P3C	RH4	0	O	DIG	LATH<4> 数据输出。
		1	I	ST	PORTH<4> 数据输入。
	AN12		I	ANA	A/D 输入通道 12。 POR 时的默认输入配置；不影响数字输出。
	P3C(2)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 C ；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RH5/AN13/P3B	RH5	0	O	DIG	LATH<5> 数据输出。
		1	I	ST	PORTH<5> 数据输入。
	AN13		I	ANA	A/D 输入通道 13。 POR 时的默认输入配置；不影响数字输出。
	P3B(2)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 B ；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RH6/AN14/P1C	RH6	0	O	DIG	LATH<6> 数据输出。
		1	I	ST	PORTH<6> 数据输入。
	AN14		I	ANA	A/D 输入通道 14。 POR 时的默认输入配置；不影响数字输出。
	P1C(2)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 C ；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RH7/AN15/P1B	RH7	0	O	DIG	LATH<7> 数据输出。
		1	I	ST	PORTH<7> 数据输入。
	AN15		I	ANA	A/D 输入通道 15。 POR 时的默认输入配置；不影响数字输出。
	P1B(2)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B ；优先于端口数据和 PSP 数据。可以在增强型 PWM关闭期间被配置为三态。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲区输入， x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 1： 在 80 引脚器件上未实现。

2： ECCPMX 配置位清零时 P1B/P1C 和 P3B/P3C 的备用分配 （80 引脚和 100 引脚器件）。默认分配是 PORTE<6:3>。

表 10-18： 与 PORTH 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTH	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	62
LATH	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	61
TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	61

10.10 PORTJ、 TRISJ 和 LATJ 寄存器

注：	PORTJ 仅在 80 引脚和 100 引脚器件上存在。

PORTJ 以两种不同方式实现为双向端口：

- 80 引脚器件：2 位宽 （RJ<5:4>）
- 100 引脚器件：8 位宽 （RJ<7:0>）

对应的数据方向寄存器是 TRISJ。将 TRISJ 某位置 1（= 1）时，会将 PORTJ 的相应引脚设为输入（即，使相应的输出驱动器呈高阻状态）。将 TRISJ 某位清零（= 0）时，会将 PORTJ 的相应引脚设为输出（即，将输出锁存器中的内容置于选中引脚）。PORTJ 上的所有引脚都是仅数字引脚，并且可以承受高达 5.5V 的电压。

数据锁存器（LATJ）也是存储器映射的。对 LATJ 寄存器执行读－修改－写操作将读写 PORTJ 的输出锁存值。

PORTJ 上的所有引脚都配有施密特触发输入缓冲区。每个引脚都可被单独配置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

当使能外部存储器接口时，所有的 PORTJ 引脚都用于控制接口的输出。通过清零 EBDIS 控制位（MEMCON<7>）使能接口时，这将自动发生。 TRISJ 位也被改写。

PORTJ 的每个引脚都具有内部弱上拉电路。提供上拉电路是为了保证上电时外部存储器接口输入为一个已知状态。单个控制位可以关闭所有上拉电路。可以通过清零 RJPU 位（PORTA<7>）来实现。当端口引脚被配置为输出时，其弱上拉电路会自动切断。发生任何器件复位时，上拉功能会被禁止。

例 10-9： 初始化 PORTJ

```
CLRF   PORTJ  ;  Initialize PORTG by
              ; clearing output
              ; data latches
CLRF   LATJ   ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0CFh   ;  Value used to
              ; initialize data
              ; direction
MOVWF  TRISJ  ;  Set RJ3:RJ0 as inputs
              ; RJ5:RJ4 as output
              ;  RJ7:RJ6 as inputs
```

 超前信息

表 **10-19：　PORTJ** 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RJ0/ALE[1]	RJ0[1]	0	O	DIG	LATJ<0> 数据输出。
		1	I	ST	PORTJ<0> 数据输入。
	ALE[1]	x	O	DIG	外部存储器接口地址锁存器使能控制输出；优先于数字 I/O。
RJ1/$\overline{OE}$[1]	RJ1[1]	0	O	DIG	LATJ<1> 数据输出。
		1	I	ST	PORTJ<1> 数据输入。
	$\overline{OE}$[1]	x	O	DIG	外部存储器接口输出使能控制输出；优先于数字 I/O。
RJ2/$\overline{WRL}$[1]	RJ2[1]	0	O	DIG	LATJ<2> 数据输出。
		1	I	ST	PORTJ<2> 数据输入。
	$\overline{WRL}$[1]	x	O	DIG	外部存储器总线写入低字节控制输出；优先于数字 I/O。
RJ3/$\overline{WRH}$[1]	RJ3[1]	0	O	DIG	LATJ<3> 数据输出。
		1	I	ST	PORTJ<3> 数据输入。
	$\overline{WRH}$[1]	x	O	DIG	外部存储器接口写入高字节控制输出；优先于数字 I/O。
RJ4/BA0	RJ4	0	O	DIG	LATJ<4> 数据输出。
		1	I	ST	PORTJ<4> 数据输入。
	BA0[2]	x	O	DIG	外部存储器接口字节地址 0 控制输出；优先于数字 I/O。
RJ5/$\overline{CE}$	RJ5	0	O	DIG	LATJ<5> 数据输出。
		1	I	ST	PORTJ<5> 数据输入。
	$\overline{CE}$[2]	x	O	DIG	外部存储器接口芯片使能控制输出；优先于数字 I/O。
RJ6/$\overline{LB}$[1]	RJ6[1]	0	O	DIG	LATJ<6> 数据输出。
		1	I	ST	PORTJ<6> 数据输入。
	$\overline{LB}$[1]	x	O	DIG	外部存储器接口低字节使能控制输出；优先于数字 I/O。
RJ7/$\overline{UB}$[1]	RJ7[1]	0	O	DIG	LATJ<7> 数据输出。
		1	I	ST	PORTJ<7> 数据输入。
	$\overline{UB}$[1]	x	O	DIG	外部存储器接口高字节使能控制输出；优先于数字 I/O。

图注：　O = 输出，I = 输入，DIG = 数字输出，ST = 施密特缓冲区输入，x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注　**1：**　仅在 100 引脚器件上实现。

　　2：　仅在 100 引脚器件上实现了 EMB 功能。

表 **10-20：　**与 **PORTJ** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTJ	RJ7[1]	RJ6[1]	RJ5	RJ4	RJ3[1]	RJ2[1]	RJ1[1]	RJ0[1]	62
LATJ	LATJ7[1]	LATJ6[1]	LATJ5	LATJ4	LATJ3[1]	LATJ2[1]	LATJ1[1]	LATJ0[1]	61
TRISJ	TRISJ7[1]	TRISJ6[1]	TRISJ5	TRISJ4	TRISJ3[1]	TRISJ2[1]	TRISJ1[1]	TRISJ0[1]	61
PORTA	RJPU	—	RA5	RA4	RA3	RA2	RA1	RA0	62

图注：　— = 未用，读为 0。PORTJ 不使用阴影单元。

注　**1：**　仅在 100 引脚器件上实现。

10.11 并行从动端口

注：	并行从动端口仅在 100 引脚器件上实现。

当控制位 PSPMODE（PSPCON<4>）置 1 时，PORTD 也可用作 8 位宽的并行从动端口或单片机端口。外部可通过 RD 控制输入引脚（RE0/AD8/RD/P2D）和 WR 控制输入引脚（RE1/AD9/WR/P2C）对其进行异步读写。

注：	并行从动端口只在单片机模式可用。

PSP 可以直接与 8 位微处理器数据总线连接。外部微处理器可以读或写 PORTD 8 位锁存值。PSPMODE 位置 1 将把端口引脚 RE0/AD8/RD/P2D 使能为 RD 输入，把 RE1/AD9/WR/P2C 使能为 WR 输入，并把 RE2/AD10/CS/P2B 使能为 CS（片选）输入。要实现此功能， TRISE 寄存器 （TRISE<2:0>）对应的数据方向位必须配置为输入 （置 1）。

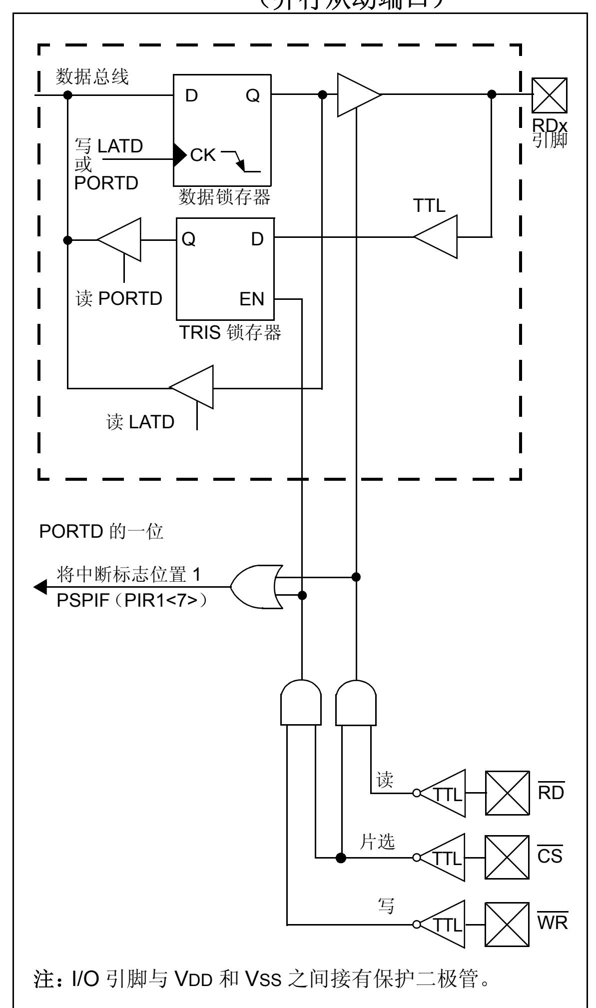
当第一次检测到 CS 和 WR 线均为低电平时发生对 PSP 的写操作，当检测到任何一根线为高电平时结束操作。写操作结束后， PSPIF 和 IBF 标志位均置 1。

当第一次检测到 CS 和 RD 线均为低电平时发生对 PSP 的读操作。PORTD 中的数据被读出且 OBF 位置 1。如果用户通过将新数据写入 PORTD 而试图将 OBF 置 1，该数据会立即被读出；但 OBF 位不会被置 1。

当 CS 或 RD 线被检测到高电平时， PORTD 引脚返回到输入状态且 PSPIF 位被置 1。用户应用程序在为 PSP 提供服务之前应该等待 PSPIF 被置 1。发生这种情况时，可以查询 IBF 和 OBF 位并进行相应的操作。

写和读模式下控制位的时序分别如图 10-3 和图 10-4 所示。

图 10-2： PORTD 和 PORTE 框图（并行从动端口）



 超前信息

寄存器 **10-1**： **PSPCON**：并行从动端口控制寄存器

R-0	R-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
IBF	OBF	IBOV	PSPMODE	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **IBF**：输入缓冲区满状态位
1 = 已接收一个字，等待 CPU 读取
0 = 未接收到任何字

bit 6 **OBF**：输出缓冲区满状态位
1 = 输出缓冲区仍保存着之前写入的字
0 = 输出缓冲区已被读取

bit 5 **IBOV**：输入缓冲区溢出检测位
1 = 之前输入的字尚未被读取时发生写操作 （必须用软件清零）
0 = 未发生溢出

bit 4 **PSPMODE**：并行从动端口模式选择位
1 = 并行从动端口模式
0 = 通用 I/O 模式

bit 3-0 未用：读为 0

图 **10-3**： 并行从动端口写波形图

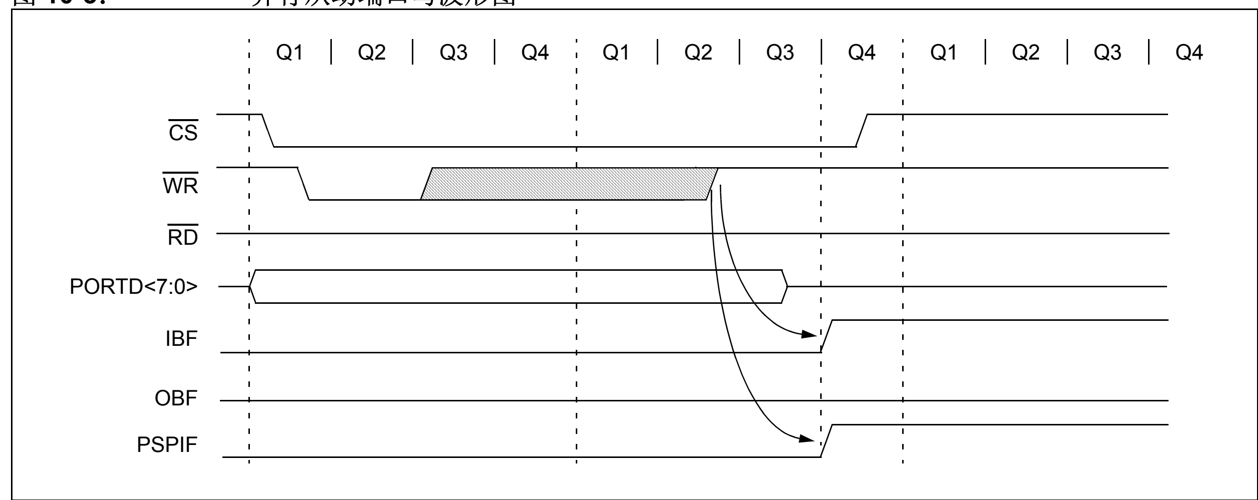


图 10-4：　　并行从动端口读波形图

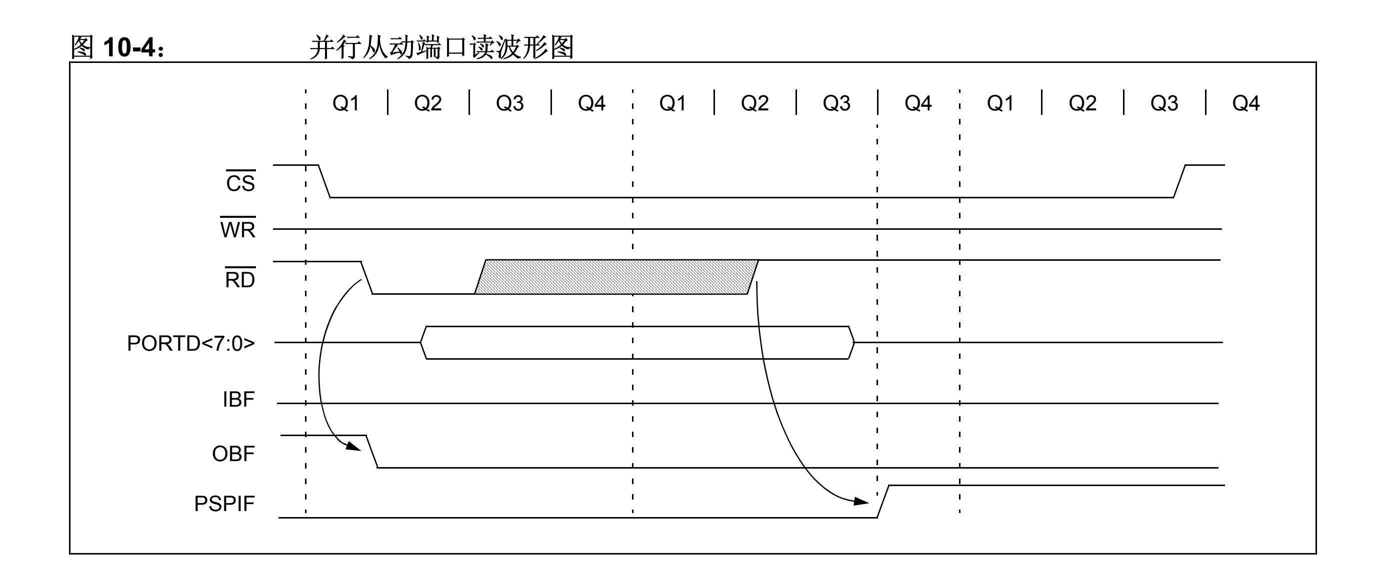


表 10-21：　　与并行从动端口相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	62
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	62
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	61
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	62
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	62
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	61
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	61
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61

图注：　　— = 未用，读为 0。并行从动端口不使用阴影单元。

11.0 TIMER0 模块

Timer0 模块具有以下特性：

- 可由软件选择作为 8 位或 16 位定时器 / 计数器
- 可读写寄存器
- 专用的 8 位软件可编程预分频器
- 可选的时钟源 （内部或外部）
- 外部时钟的边沿选择
- 溢出时中断

T0CON 寄存器 （寄存器 11-1）控制该模块操作的所有方面，包括预分频比的选择。它是可读写的。

图 11-1 给出了 8 位模式下 Timer0 模块的简化框图。图 11-2 给出了16 位模式下 Timer0 模块的简化框图。

寄存器 11-1： T0CON：TIMER0 控制寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **TMR0ON**：Timer0 开 / 关控制位
1 = 使能 Timer0
0 = 停止 Timer0

bit 6 **T08BIT**：Timer0 8 位 /16 位控制位
1 = Timer0 被配置为 8 位定时器 / 计数器
0 = Timer0 被配置为 16 位定时器 / 计数器

bit 5 **T0CS**：Timer0 时钟源选择位
1 = T0CKI 引脚上的传输信号
0 = 内部指令周期时钟 （CLKO）

bit 4 **T0SE**：Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚上电平的下降沿递增
0 = 在 T0CKI 引脚上电平的上升沿递增

bit 3 **PSA**：Timer0 预分频器分配位
1 = 未分配 Timer0 预分频器。 Timer0 时钟输入不经过预分频器。
0 = 已分配 Timer0 预分频器。 Timer0 时钟输入来自预分频器的输出。

bit 2-0 **T0PS2:T0PS0**：Timer0 预分频值选择位
111 = 1:256 预分频值
110 = 1:128 预分频值
101 = 1:64 预分频值
100 = 1:32 预分频值
011 = 1:16 预分频值
010 = 1:8 预分频值
001 = 1:4 预分频值
000 = 1:2 预分频值

11.1 Timer0 工作原理

Timer0 既可用作定时器也可用作计数器。可通过 T0CS 位（T0CON<5>）来选择模式。在定时器模式下（T0CS = 0），该模块在每个时钟周期计时都会递增（默认情况下），除非选择了其他预分频值（见**第 11.3 节“预分频器”**）。如果写入 TMR0，那么在随后的两个指令周期内，计时都不再递增。用户可通过将调整值写入 TMR0 寄存器来避开这一问题。

通过将 T0CS 位置 1（= 1）选择计数器模式。在该模式下，Timer0 可在 RA4/T0CKI 引脚上信号的每个上升沿或下降沿递增。递增边沿由 Timer0 时钟源边沿选择位 T0SE（T0CON<4>）决定。清零该位即选择上升沿。下面讨论外部时钟输入的限制条件。

可以使用外部时钟源来驱动 Timer0。但是，必须满足一定要求，以确保外部时钟和内部相位时钟（TOSC）保持同步。在同步之后，定时器 / 计数器仍需要一定的延时才会引发递增操作。

11.2 Timer0 的 16 位读写模式

TMR0H 并不是 16 位模式下 Timer0 的高字节，而是被缓存的 Timer0 高字节。Timer0 的高字节不可以被直接读写（见图 11-2）。在读 TMR0L 时使用 Timer0 高字节的内容更新 TMR0H。这样可以一次读取 Timer0 的全部 16 位，而无需验证读到的高字节和低字节的有效性（在连续读取高字节和低字节时，由于可能存在进位，因此需要验证读到的高字节和低字节的有效性）。

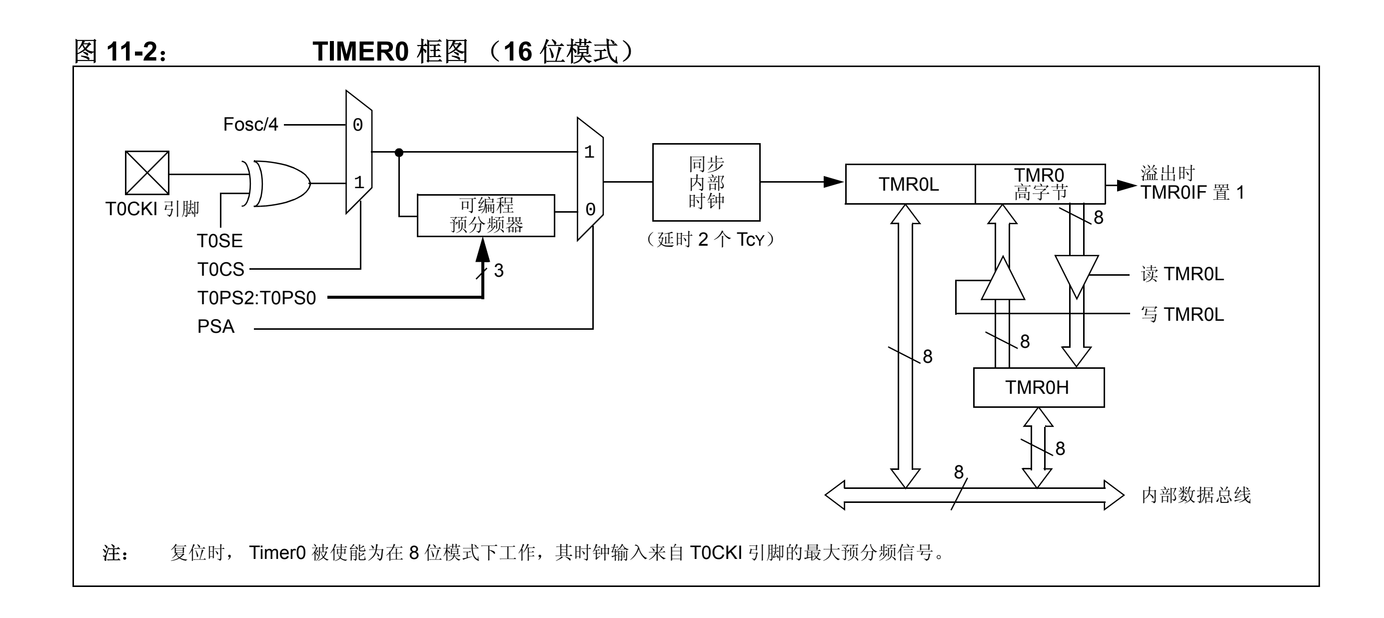
同样，写入 Timer0 的高字节也是通过 TMR0H 缓冲寄存器来操作的。在写入 TMR0L 的同时，使用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

图 11-1： TIMER0 框图（8 位模式）

FOSC/4
T0CKI 引脚
T0SE
T0CS
T0PS2:T0PS0
PSA
可编程预分频器
同步内部时钟
（延时 2 个 TCY）
TMR0L
溢出时 TMR0IF 置 1
内部数据总线

注： 复位时， Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

图 11-2： TIMER0 框图（16 位模式）



 超前信息

11.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。该预分频器不可直接读写。通过 PSA 和 T0PS2:T0PS0 位（T0CON<3:0>）进行预分频器的分配和设定预分频比值。

将 PSA 位清零可将预分频器分配给 Timer0 模块。预分频值可以在 1:2 到 1:256 之间进行选择，以 2 的整数次幂递增。

如果将预分频器分配给 Timer0 模块，所有写入 TMR0 寄存器的指令（例如，`CLRF TMR0`、`MOVWF TMR0` 和 `BSF TMR0` 等），都会将预分频器的计数值清零。

> 注： 如果将预分频器分配给 Timer0，写入 TMR0 会将预分频器的计数值清零，但不会改变预分频器的分配。

11.3.1 切换预分频器的分配

预分频器的分配完全由软件控制，并且在程序执行期间可以随时更改。

11.4 Timer0 中断

8 位模式下的 TMR0 寄存器从 FFh 到 00h 发生溢出，或 16 位模式下的 TMR0 从 FFFFh 到 0000h 发生溢出时，将产生 TMR0 中断。这种溢出会使 TMR0IF 标志位置 1。可以通过清零 TMR0IE 位（INTCON<5>）来屏蔽该中断。在重新允许该中断前，必须在中断服务程序中用软件清零 TMR0IF 位。

由于 Timer0 在休眠模式下是关闭的，所以 TMR0 中断无法将处理器从休眠状态唤醒。

表 11-1： 与 TIMER0 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TMR0L	Timer0 寄存器的低字节								60
TMR0H	Timer0 寄存器的高字节								60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	59
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	60
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	61

图注： — = 未用，读为 0。 Timer0 不使用阴影单元。

注：

12.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下特性：

- 可由软件选择作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR1H 和 TMR1L）
- 可选择器件时钟或 Timer1 内部振荡器作为时钟源 （内部或外部）
- 溢出时中断
- ECCP 特殊事件触发模块复位
- 器件时钟状态标志位 （T1RUN）

图 12-1 给出了 Timer1 模块的简化框图。图 12-2 给出了此模块在读写模式下的工作原理框图。

此模块自身带有低功耗振荡器可提供额外的时钟选项。Timer1 振荡器也可作为单片机处于节能状态时的低功耗时钟源。

仅需极少量外部元件和代码开销，Timer1 就可为应用提供实时时钟 （RTC）。

Timer1 由 T1CON 控制寄存器 （寄存器 12-1）控制。该寄存器还包含 Timer1 振荡器使能位 （T1OSCEN）。可以通过将控制位 TMR1ON （T1CON<0>）置 1 或清零来使能或禁止 Timer1。

寄存器 12-1： T1CON：TIMER1 控制寄存器

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能 Timer1 通过一次 16 位操作进行寄存器读 / 写
0 = 使能 Timer1 通过两次 8 位操作进行寄存器读 / 写

bit 6 **T1RUN**：Timer1 系统时钟状态位
1 = 器件时钟由 Timer1 振荡器产生
0 = 器件时钟由另一个时钟源产生

bit 5-4 **T1CKPS1:T1CKPS0**：Timer1 输入时钟预分频值选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 3 **T1OSCEN**：Timer1 振荡器使能位
1 = 使能 Timer1 振荡器
0 = 关闭 Timer1 振荡器
关闭振荡器的反相器和反馈电阻以降低功耗。

bit 2 **$\overline{\text{T1SYNC}}$**：Timer1 外部时钟输入同步选择位
当 TMR1CS = 1 时：
1 = 不同步外部时钟输入
0 = 同步外部时钟输入
当 TMR1CS = 0 时：
该位为无关位。当 TMR1CS = 0 时， Timer1 使用内部时钟。

bit 1 **TMR1CS**：Timer1 时钟源选择位
1 = 使用 RC0/T1OSO/T13CKI 引脚上的外部时钟 （上升沿计数）
0 = 内部时钟 （Fosc/4）

bit 0 **TMR1ON**：Timer1 使能位
1 = 使能 Timer1
0 = 停止 Timer1

12.1 Timer1 工作原理

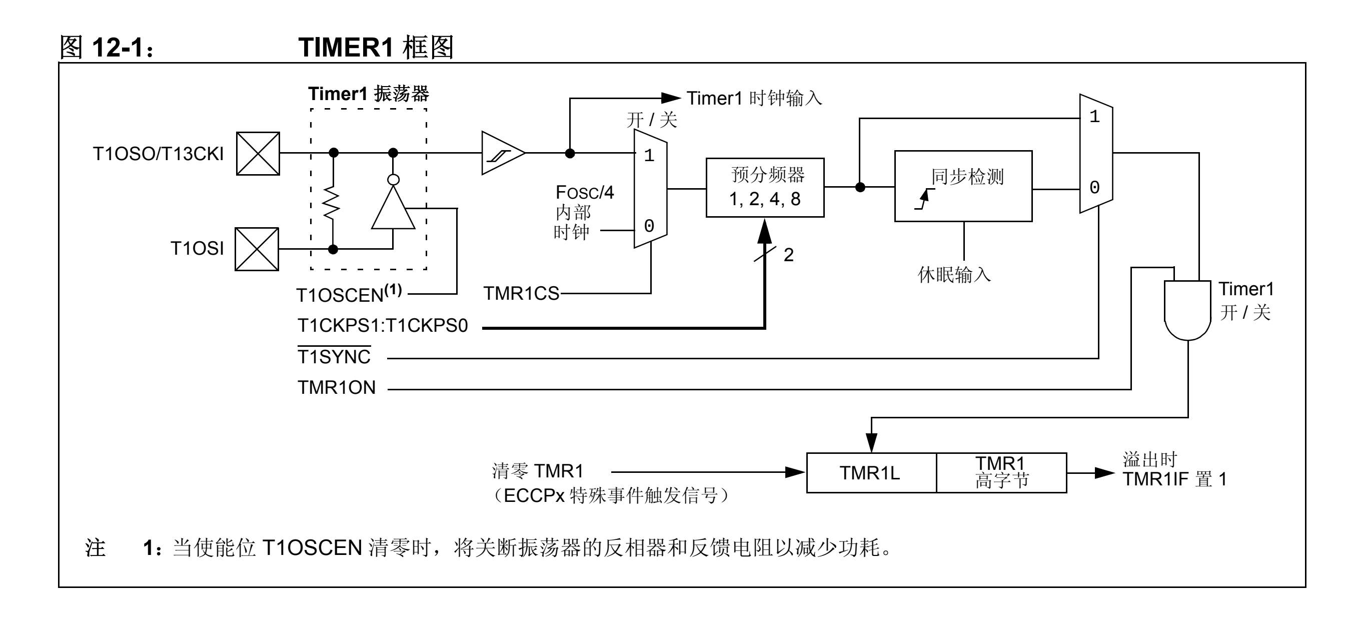
Timer1 可在以下模式工作：

- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR1CS（T1CON<1>）决定。当 TMR1CS 清零 （= 0）时， Timer1 在每个内部指令周期（Fosc/4）递增。当该位置 1 时，Timer1 在 Timer1 外部时钟输入信号或 Timer1 振荡器输出信号 （如果使能）的每个上升沿递增。

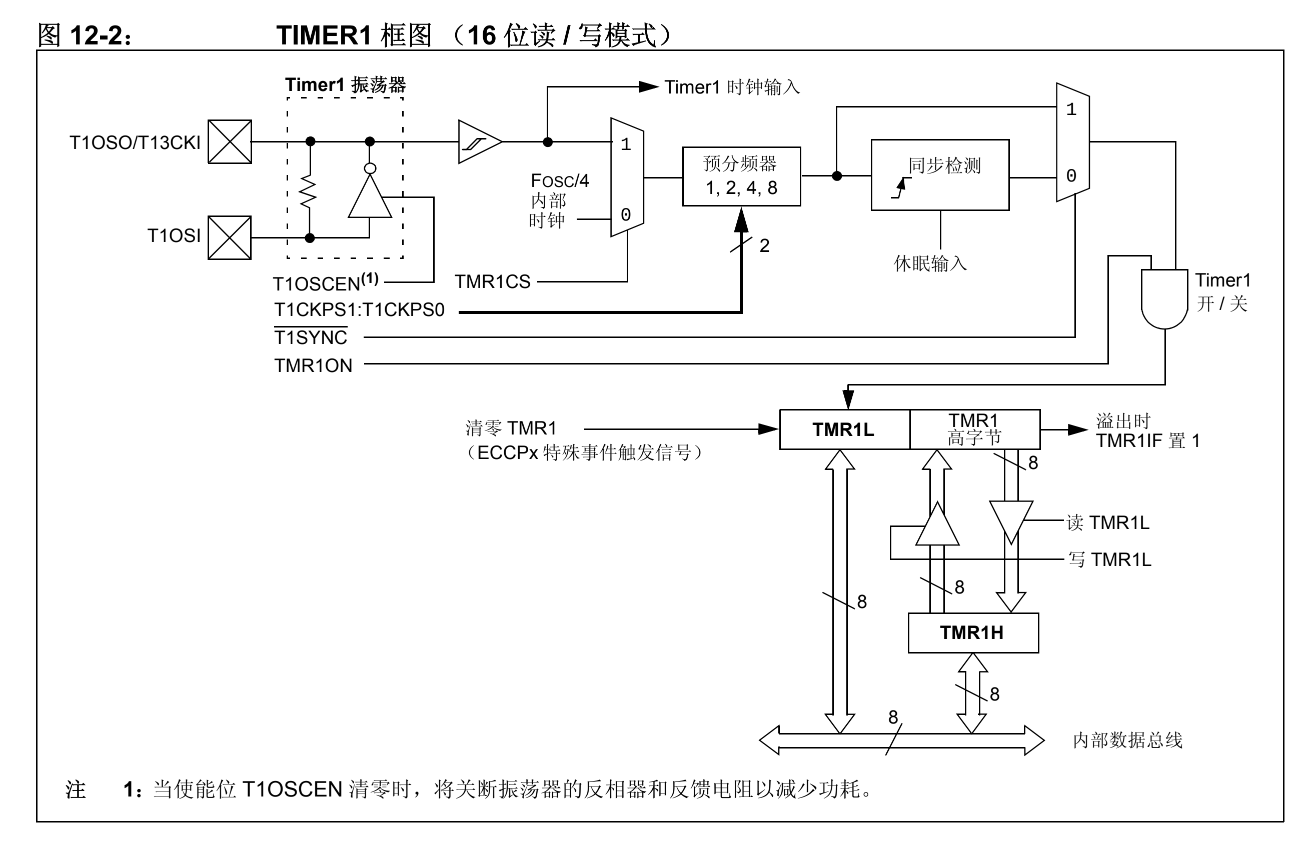
当使能 Timer1 时，RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 12-1： TIMER1 框图



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以减少功耗。

图 12-2： TIMER1 框图 （16 位读 / 写模式）



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以减少功耗。

 超前信息

12.2 Timer1 的 16 位读 / 写模式

可将 Timer1 配置为 16 位读写模式（见图 12-2）。当 RD16 控制位（T1CON<7>）置 1 时，TMR1H 的地址被映射到 Timer1 的高字节缓冲寄存器。读 TMR1L 将把 Timer1的高字节的内容装入Timer1高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer1 的全部 16 位，而不需要像先读高字节再读低字节那样，由于两次读取之间可能存在进位，而不得不验证读取的有效性。

写 Timer1 的高字节也必须通过 TMR1H 缓冲寄存器进行。在写入 TMR1L 的同时，使用 TMR1H 的内容更新 Timer1 的高字节。这样允许用户将 16 位值一次写入 Timer1 的高字节和低字节。

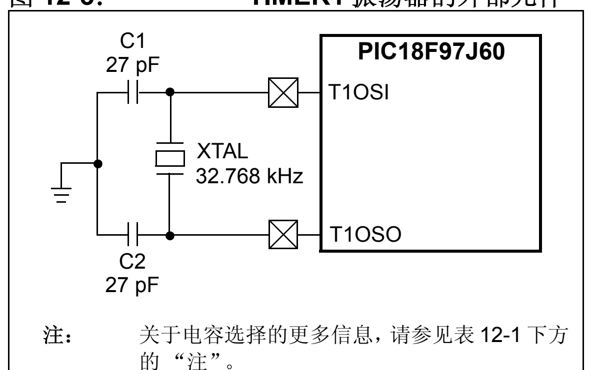
在该模式下不能直接读写 Timer1 的高字节。所有读写都必须通过 Timer1 高字节缓冲寄存器来进行。写入 TMR1H 不会清零 Timer1 预分频器。只有在写 TMR1L 时才会清零该预分频器。

12.3 Timer1 振荡器

片上晶体振荡器电路连接在 T1OSI（输入）引脚和 T1OSO（放大器输出）引脚之间。可以通过将 Timer1 振荡器使能位 T1OSCEN（T1CON<3>）置 1 来使能该振荡电路。该振荡电路是一种低功耗电路，它采用了额定振荡频率为 32 kHz 的晶振。在所有功耗管理模式下都可继续运行。图 12-3 所示是典型的 LP 振荡器电路。表 12-1 给出了供 Timer1 振荡器选择的电容值。

用户必须提供软件延时来确保 Timer1 振荡器的正常起振。

图 12-3： TIMER1 振荡器的外部元件



注： 关于电容选择的更多信息，请参见表 12-1 下方的“注”。

表 12-1： TIMER1振荡器的电容选择[2,3,4]

振荡器类型	频率	C1	C2
LP	32 kHz	27 pF[1]	27 pF[1]

注 1：Microchip 建议将该值作为验证振荡电路的起始点。
2：电容越大，振荡器越稳定，但起振时间越长。
3：因为每种谐振器 / 晶振都有其自身特性，用户应当向谐振器 / 晶振制造厂商咨询外部元件的适当值。
4：上述电容值仅供设计参考。

12.3.1 使用 TIMER1 作为时钟源

在功耗管理模式下也可以将Timer1振荡器用作时钟源。通过将时钟选择位 SCS1:SCS0（OSCCON<1:0>）设置为 01，器件可以切换到 SEC_RUN 模式，CPU 和外设都可以用 Timer1 振荡器作为时钟源。如果 IDLEN 位（OSCCON<7>）被清零并且执行了 SLEEP 指令，器件将进入 SEC_IDLE 模式。更多详细信息，请参见**第 3.0 节“功耗管理模式”**。

无论何时将 Timer1 振荡器用作时钟源，Timer1 系统时钟状态标志位 T1RUN（T1CON<6>）均会置 1。这可用于确定控制器的当前时钟模式。该位也可指示故障保护时钟监视器当前正使用的时钟源。如果使能了故障保护时钟监视器并且 Timer1 振荡器在提供时钟信号时发生了故障，查询 T1RUN 位可以确定时钟源是 Timer1 振荡器还是其他时钟源。

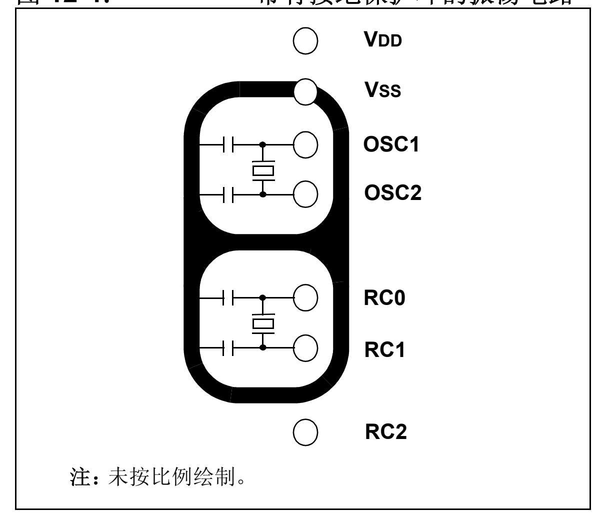
12.3.2 TIMER1 振荡器布线注意事项

Timer1 振荡器电路在工作期间仅消耗极少的电流。鉴于此振荡器的低功耗特性，它对附近变化较快的信号比较敏感。

如图 12-3 所示，振荡电路应该尽可能靠近单片机。除了 VSS 或 VDD 外，在该振荡电路区域内不应有其他电路。

如果必须要在该振荡器附近布置高速电路（如输出比较模式或 PWM 模式的 ECCP1 引脚，或使用 OSC2 引脚的主振荡器），那么在该振荡电路周围布置接地保护环（如图 12-4 所示），对于单面 PCB 板或外加接地层的电路板来讲可能会有帮助。

图 12-4： 带有接地保护环的振荡电路



12.4 Timer1 中断

TMR1 寄存器对（TMR1H:TMR1L）从 0000h 开始，增加到 FFFFh，然后溢出返回到 0000h 重新开始计数。如果允许了 Timer1 中断，则溢出时会产生 Timer1 中断，并由中断标志位 TMR1IF （PIR1<0>）捕捉。可以通过对 Timer1 中断允许位 TMR1IE （PIE1<0>）置 1 或清零来允许或禁止该中断。

12.5 使用 ECCP 特殊事件触发信号复位 Timer1

如果 ECCP1 或 ECCP2 配置为在比较模式下产生特殊事件触发信号 （CCPxM3:CCPxM0 = `1011`），该信号将复位 Timer3。如果使能了 A/D 模块，来自 ECCP2 的触发信号还将启动 A/D 转换 （更多信息，请参见**第 17.2.1 节 "特殊事件触发器"**）。

要使用这一功能，必须将模块配置为定时器或同步计数器。在这种情况下，CCPRxH:CCPRxL 这对寄存器实际上变成了 Timer1 的周期寄存器。

如果 Timer1 在异步计数器模式下运行，复位操作可能不起作用。

如果 Timer1 的写操作和特殊事件触发同时发生，则写操作优先。

注：	ECCPx 模块产生的特殊事件触发信号不会将 TMR1IF 中断标志位（PIR1<0>）置 1。

12.6 使用 Timer1 作为实时时钟

为 Timer1 外接一个 LP 振荡器（如**第 12.3 节"Timer1 振荡器"**中所述），可以允许用户在他们的应用中包括 RTC 功能。只需通过一个提供精确时基的廉价时钟晶振以及几行计算时间的应用程序代码就可以实现这一功能。当器件在休眠模式下工作并使用电池或超大容量电容作为电源时，可省去另外的 RTC 器件和备用电池。

应用代码程序 `RTCisr`（如例 12-1 所示），演示了使用中断服务程序以 1 秒的间隔递增计数器的简单方法。将 TMR1 寄存器对的值递增至溢出将触发中断并调用中断服务程序，该程序会使秒计数器加 1，其他的分钟和小时计数器则会在前面的计数器溢出时加 1。

由于这对寄存器为 16 位宽，因此使用 32.768 kHz 时钟，将其计数到溢出需要 2 秒。要使溢出按所需的 1 秒间隔进行，必须预先装载这对寄存器。最简单的方法是使用 BSF 指令将 TMR1H 的最高有效位置 1。请注意决不要预先加载或改变 TMR1L 寄存器，这样做可能会引起多个周期的累积误差。

要使此方法精确，Timer1 必须工作于异步模式且必须允许 Timer1 溢出中断（PIE1<0> = `1`），如程序 `RTCinit` 所示。同时 Timer1 振荡器也必须被使能并始终运行。

 超前信息

例 **12-1：** 使用 **TIMER1** 中断服务实现实时时钟

```
RTCinit
        MOVLW   80h             ; Preload TMR1 register pair
        MOVWF   TMR1H           ; for 1 second overflow
        CLRF    TMR1L
        MOVLW   b'00001111'     ; Configure for external clock,
        MOVWF   T1CON           ; Asynchronous operation, external oscillator
        CLRF    secs            ; Initialize timekeeping registers
        CLRF    mins            ;
        MOVLW   .12
        MOVWF   hours
        BSF     PIE1, TMR1IE    ; Enable Timer1 interrupt
        RETURN
RTCisr
        BSF     TMR1H, 7        ; Preload for 1 sec overflow
        BCF     PIR1, TMR1IF    ; Clear interrupt flag
        INCF    secs, F         ; Increment seconds
        MOVLW   .59             ; 60 seconds elapsed?
        CPFSGT  secs
        RETFIE              ; No, done
        CLRF    secs            ; Clear seconds
        INCF    mins, F         ; Increment minutes
        MOVLW   .59             ; 60 minutes elapsed?
        CPFSGT  mins
        RETFIE                  ; No, done
        CLRF    mins            ; clear minutes
        INCF    hours, F        ; Increment hours
        MOVLW   .23             ; 24 hours elapsed?
        CPFSGT  hours
        RETFIE                  ; No, done
        CLRF    hours           ; Reset hours
        RETFIE                  ; Done
```

表 **12-2：** 与 **TIMER1** 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
TMR1L	Timer1 寄存器的低字节								60
TMR1H	Timer1 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	60

图注： Timer1 模块不使用阴影单元。

注：

13.0 TIMER2 模块

Timer2 定时器模块具有以下特性：

- 8 位定时器和周期寄存器（分别为 TMR2 和 PR2）
- 可读写 （以上两个寄存器）
- 可软件编程的预分频器 （分频比为 1:1、 1:4 和 1:16）
- 可软件编程的后分频器 （分频比为 1:1 到 1:16）
- TMR2 与 PR2 匹配时产生中断
- 可选作 MSSP 模块的移位时钟

此模块由 T2CON 寄存器（寄存器 13-1）控制，此寄存器使能或禁止定时器并配置预分频器和后分频器。可以通过清零控制位 TMR2ON（T2CON<2>）关闭 Timer2，以实现功耗最小。

图 13-1 给出了此模块的简化框图。

13.1 Timer2 工作原理

在正常工作模式下， TMR2 从 00h 开始，每个时钟周期（Fosc/4）加 1。4 位计数器 / 预分频器提供了对时钟输入不分频、4 分频和 16 分频三种选项，并可通过预分频控制位 T2CKPS1:T2CKPS0 （T2CON<1:0>）进行选择。在每个时钟周期，TMR2的值都会与周期寄存器PR2中的值进行比较。当两个值匹配时，由比较器产生匹配信号作为定时器的输出。此信号也会使 TMR2 的值在下一个周期复位到 00h，并驱动计数器 / 后分频器 （见**第 13.2 节 "Timer2 中断"**）。

TMR2 和 PR2 寄存器均可直接读写。在任何器件复位时，TMR2 寄存器都会清零，而 PR2 寄存器则初始化为 FFh。预分频和后分频计数器均会在发生以下事件时清零：

- 对 TMR2 寄存器进行写操作
- 对 T2CON 寄存器进行写操作
- 任何器件复位 （上电复位、 $\overline{\text{MCLR}}$ 复位、看门狗定时器复位或欠压复位）

写 T2CON 时 TMR2 不会清零。

寄存器 13-1： T2CON：TIMER2 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7　未用：读为 0

bit 6-3　**T2OUTPS3:T2OUTPS0**：Timer2 输出后分频比选择位
　0000 = 1:1 后分频比
　0001 = 1:2 后分频比
　•
　•
　•
　1111 = 1:16 后分频比

bit 2　**TMR2ON**：Timer2 使能位
　1 = 使能 Timer2
　0 = 关闭 Timer2

bit 1-0　**T2CKPS1:T2CKPS0**：Timer2 时钟预分频值选择位
　00 = 预分频值为 1
　01 = 预分频值为 4
　1x = 预分频值为 16

13.2 Timer2 中断

Timer2 也可以产生可选的器件中断。 Timer2 输出信号（TMR2 与 PR2 匹配时）为 4 位输出计数器 / 后分频器提供输入。此计数器产生的 TMR2 匹配中断标志位为 TMR2IF （PIR1<1>）。可以通过将 TMR2 匹配中断允许位 TMR2IE （PIE1<1>）置 1 来允许此中断。

可以通过后分频控制位 T2OUTPS3:T2OUTPS0 （T2CON<6:3>）在 16 个后分频比值选项 （从 1:1 到 1:16）中选择其一。

13.3 Timer2 输出

TMR2 的不经分频的输出主要用于 CCP 模块，它用作 CCP 模块在 PWM 模式下工作时的时基。

还可选择将 Timer2 用作 MSSP 模块在 SPI 模式下工作时的移位时钟源。**第 19.0节"主控同步串行口（MSSP）模块"** 中提供了更多信息。

图 13-1： TIMER2 框图

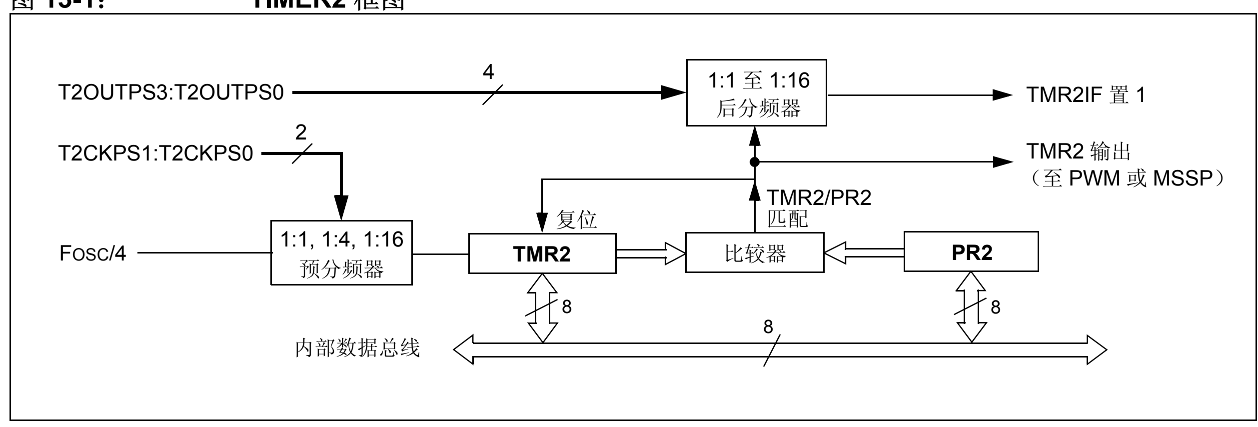


表 13-1： 与 TIMER2 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
TMR2	Timer2 寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
PR2	Timer2 周期寄存器								60

图注： — = 未用，读为 0。 Timer2 模块不使用阴影单元。

 超前信息

14.0 TIMER3 模块

Timer3 定时器 / 计数器模块具有以下特性：

- 可由软件选择作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR3H 和 TMR3L）
- 可选择器件时钟或 Timer1 内部振荡器作为时钟源 （内部或外部）
- 溢出时中断
- 可利用 CCP 特殊事件触发模块复位

图 14-1 给出了 Timer3 模块的简化框图。图 14-2 给出了此模块在读写模式下的工作原理框图。

Timer3 模块是通过 T3CON 寄存器 （寄存器 14-1）来控制的。此寄存器还可用作 CCP 和 ECCP 模块的可选时钟源。更多信息，请参见**第 16.1.1 节 “CCP 模块和定时器资源”**。

寄存器 14-1： T3CON：TIMER3 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能 Timer3 通过一次 16 位操作进行寄存器读 / 写
0 = 使能 Timer3 通过两次 8 位操作进行寄存器读 / 写

bit 6,3 **T3CCP2:T3CCP1**：CCPx 的时钟源 （Timer3 和 Timer1）使能位
11 = Timer3 和 Timer4 作为所有 CCP/ECCP 模块的时钟源
10 = Timer3 和 Timer4 作为 ECCP3、 CCP4 和 CCP5 的时钟源；
Timer1 和 Timer2 作为 ECCP1 和 ECCP2 的时钟源
01 = Timer3 和 Timer4 作为 ECCP2、 ECCP3、 CCP4 和 CCP5 的时钟源；
Timer1 和 Timer2 作为 ECCP1 的时钟源
00 = Timer1 和 Timer2 作为所有 CCP/ECCP 模块的时钟源

bit 5-4 **T3CKPS1:T3CKPS0**：Timer3 输入时钟预分频值选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 2 **$\overline{\text{T3SYNC}}$**：Timer3 外部时钟输入同步选择位 （不适用于器件时钟来自 Timer1/Timer3 的场合。）
当 TMR3CS = 1 时：
1 = 不同步外部时钟输入
0 = 同步外部时钟输入
当 TMR3CS = 0 时：
该位为无关位。当 TMR3CS = 0 时， Timer3 使用内部时钟。

bit 1 **TMR3CS**：Timer3 时钟源选择位
1 = 使用Timer1振荡器或T13CKI引脚信号作为外部时钟输入（在第一个下降沿之后的上升沿开始计数）
0 = 内部时钟 （FOSC/4）

bit 0 **TMR3ON**：Timer3 使能位
1 = 使能 Timer3
0 = 停止 Timer3

14.1 Timer3 工作原理

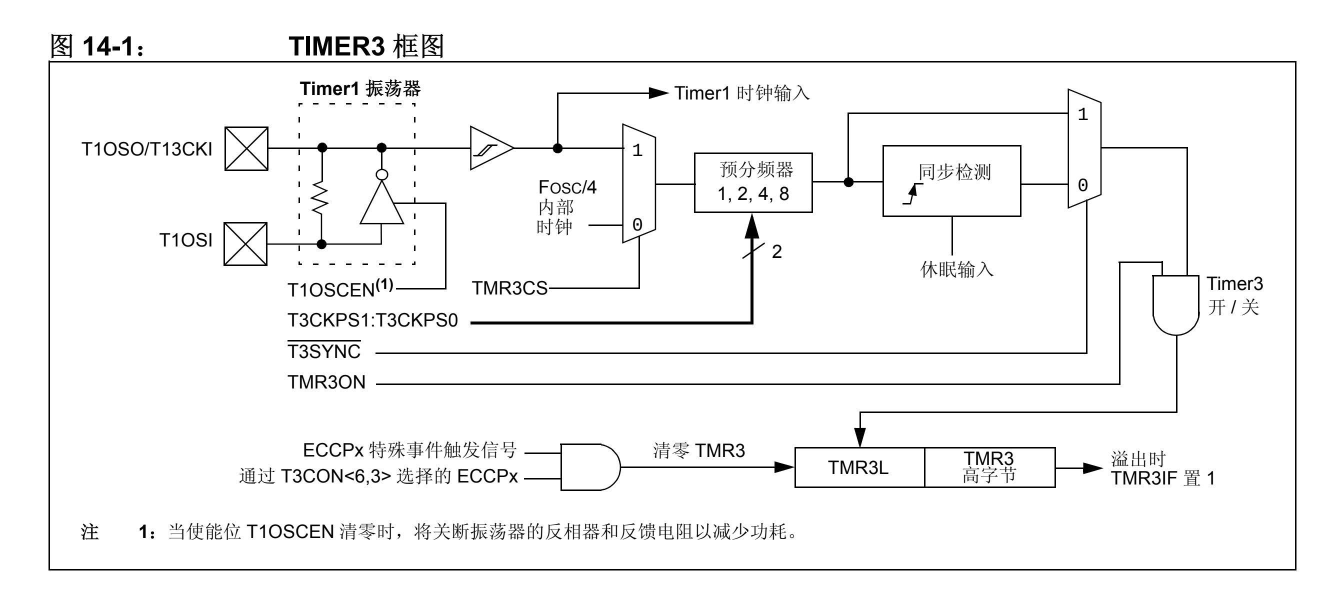
Timer3 可工作在以下三种模式之一：

- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR3CS（T3CON<1>）决定。当 TMR3CS 清零 （= 0）时， Timer3 在每个内部指令周期（Fosc/4）递增。当该位置 1 时，Timer3 在 Timer1 外部时钟输入信号或 Timer1 振荡器输出信号 （如果使能）的每个上升沿递增。

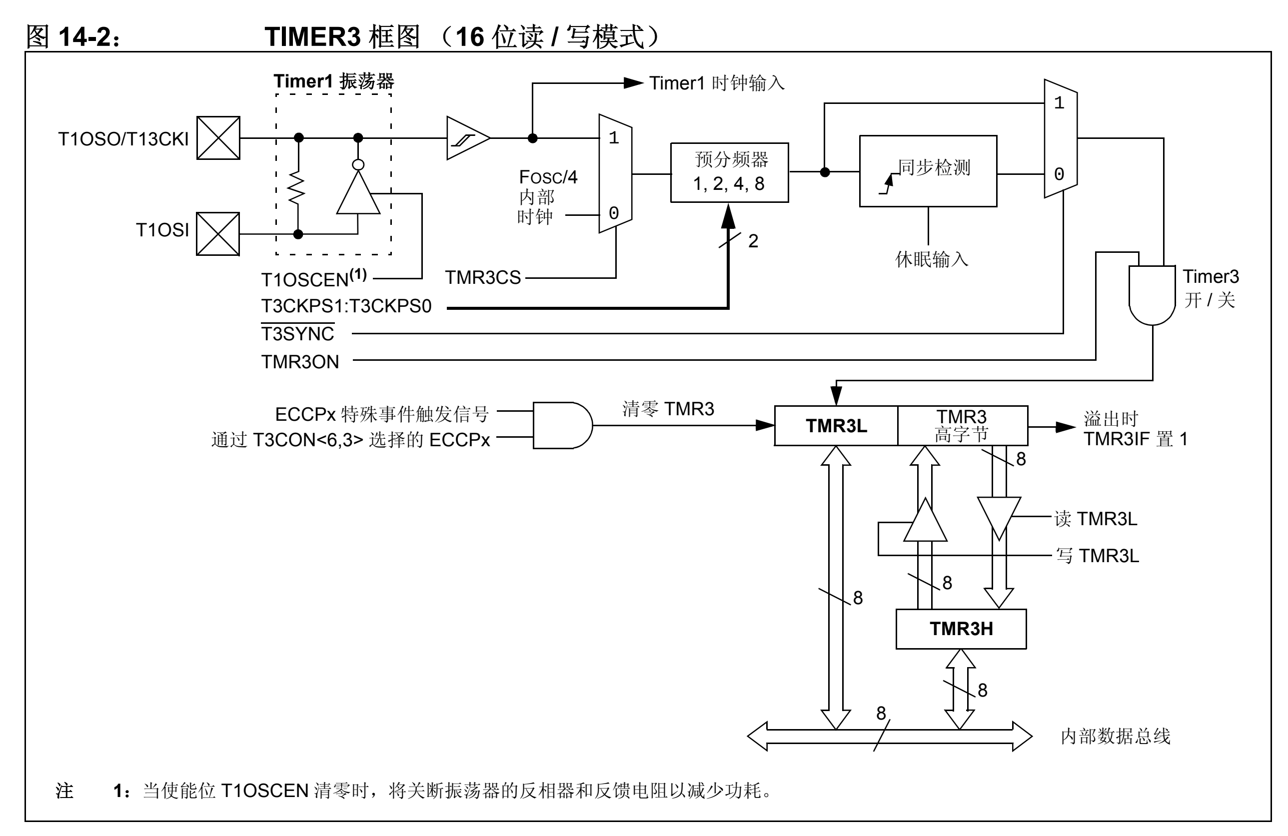
当使能 Timer1 时，RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 14-1： TIMER3 框图



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以减少功耗。

图 14-2： TIMER3 框图 （16 位读 / 写模式）



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以减少功耗。

 超前信息

14.2 Timer3 16 位读 / 写模式

可将 Timer3 配置为 16 位读写模式（见图 14-2）。当 RD16 控制位（T3CON<7>）置 1 时，TMR3H 的地址被映射到 Timer3 的高字节缓冲寄存器。读 TMR3L 将把 Timer3的高字节的内容装入Timer3高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer1 的全部 16 位，而不需要像先读高字节再读低字节那样，由于两次读取之间可能存在进位，而不得不验证读取的有效性。

写 Timer3 的高字节也必须通过 TMR3H 缓冲寄存器进行。在写入 TMR3L 的同时，使用 TMR3H 的内容更新 Timer3 的高字节。这样允许用户将 16 位值一次写入 Timer3 的高字节和低字节。

在该模式下不能直接读写 Timer3 的高字节。所有读写都必须通过 Timer3 高字节缓冲寄存器来进行。

写入 TMR3H 不会清零 Timer3 预分频器。只有在写 TMR3L 时才会清零该预分频器。

14.3 使用 Timer1 振荡器作为 Timer3 的时钟源

Timer1 内部振荡器可用作 Timer3 的时钟源。通过将 T1OSCEN（T1CON<3>）位置 1，可使能 Timer1 振荡器。要将它用作 Timer3 的时钟源，还必须将 TMR3CS 位置 1。如前文所述，这样做也会将 Timer3 配置为在振荡器的每个上升沿递增。

在**第 12.0 节 “Timer1 模块”**中对 Timer1 振荡器进行了描述。

14.4 Timer3 中断

TMR3 寄存器对（TMR3H:TMR3L）从 0000h 开始，增加到 FFFFh，然后溢出返回到 0000h 重新开始计数。如果允许了 Timer3 中断，则溢出时会产生 Timer3 中断，并由中断标志位 TMR3IF （PIR2<1>）捕捉。可以通过对 Timer3 中断允许位 TMR3IE （PIE2<1>）置 1 或清零来允许或禁止该中断。

14.5 使用 ECCP 特殊事件触发信号复位 Timer3

如果 ECCP1 或 ECCP2 配置为使用 Timer3，并在比较模式下产生特殊事件触发信号（CCPxM3:CCPxM0 = `1011`），该信号将复位 Timer3。如果使能了 A/D 模块，来自 ECCP2 的触发信号还将启动 A/D 转换（更多信息，请参见**第 17.2.1 节 “特殊事件触发器”**）。

要使用这一功能，必须将模块配置为定时器或同步计数器。在这种情况下，CCPRxH:CCPRxL 这对寄存器实际上变成了 Timer3 的周期寄存器。

如果 Timer3 在异步计数器模式下运行，复位操作可能不起作用。

如果 Timer3 的写操作和来自 ECCP 模块的特殊事件同时发生，则写操作优先。

注： ECCPx 模块产生的特殊事件触发信号不会将 TMR3IF 中断标志位（PIR2<1>）置 1。

表 14-1： 与 TIMER3 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	61
TMR3L	Timer3 寄存器的低字节								60
TMR3H	Timer3 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	60

图注： — = 未用（读为 `0`），r = 保留。Timer3 模块不使用阴影单元。

注：

15.0 TIMER4 模块

Timer4 模块具有以下特性：

- 8 位定时器寄存器 （TMR4）
- 8 位周期寄存器 （PR4）
- 可读写 （以上两个寄存器）
- 软件可编程预分频器（分频比为 1:1、1:4 和 1:16）
- 软件可编程后分频器 （分频比为 1:1 到 1:16）
- TMR4 与 PR4 匹配时产生中断

Timer4 具有一个控制寄存器，如寄存器 15-1 所示。可以通过清零控制位 TMR4ON （T4CON<2>）关闭 Timer4，以实现功耗最小。此寄存器还控制对 Timer4 的预分频比和后分频比的选择。图 15-1 所示为 Timer4 模块的简化框图。

15.1 Timer4 工作原理

Timer4 可以作为 CCP 模块在 PWM 模式下的 PWM 时基。TMR4 寄存器是可读写的，在任何器件复位时都会被清零。输入时钟（Fosc/4）有三种预分频比，分别是 1:1、 1:4 或 1:16，可通过控制位 T4CKPS1:T4CKPS0（T4CON<1:0>）选择。 TMR4 的匹配输出通过一个 4 位后分频器 （分频比在 1:1 到 1:16 之间）产生 TMR4 中断，由标志位 TMR4IF （PIR3<3>）表示。

预分频和后分频计数器在发生以下事件时均会清零：

- 对 TMR4 寄存器进行写操作
- 对 T4CON 寄存器进行写操作
- 任何器件复位 （上电复位、 $\overline{\text{MCLR}}$ 复位、看门狗定时器复位或欠压复位）

写 T4CON 时， TMR4 不会清零。

寄存器 15-1： T4CON：TIMER4 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 未用：读为 0

bit 6-3 **T4OUTPS3:T4OUTPS0**：Timer4 输出后分频比选择位
0000 = 1:1 后分频比
0001 = 1:2 后分频比
•
•
•
1111 = 1:16 后分频比

bit 2 **TMR4ON**：Timer4 使能位
1 = 使能 Timer4
0 = 关闭 Timer4

bit 1-0 **T4CKPS1:T4CKPS0**：Timer4 时钟预分频值选择位
00 = 预分频值为 1
01 = 预分频值为 4
1x = 预分频值为 16

15.2 Timer4 中断

Timer4 模块具有一个 8 位周期寄存器 PR4，该寄存器是可读写的。 Timer4 从 00h 开始递增，直到与 PR4 匹配为止，然后在下一个计数周期复位为 00h。在复位时，PR4 寄存器初始化为 FFh。

15.3 TMR4 的输出

TMR4 的输出 （在通过后分频器之前）只能用作 CCP 模块的 PWM 时基。它不可以像 Timer2 输出一样用作 MSSP 模块的波特率时钟。

图 15-1： TIMER4 框图

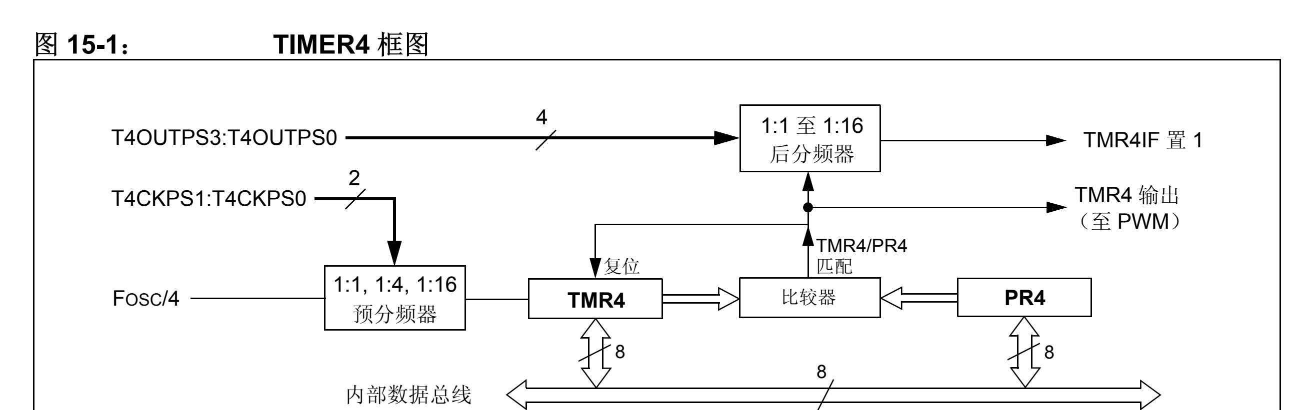


表 15-1： 与 TIMER4 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
TMR4	Timer4 寄存器								62
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	62
PR4	Timer4 周期寄存器								62

图注： — = 未用，读为 0。 Timer4 模块不使用阴影单元。

 超前信息

16.0 捕捉 / 比较 /PWM （CCP）模块

PIC18F97J60系列器件都有5个CCP（捕捉/比较/PWM）模块，其中两个模块（CCP4 和 CCP5）实现标准的捕捉、比较和脉宽调制（Pulse-Width Modulation， PWM）模式，在本节中将讨论这两个模块。另外三个模块（ECCP1、ECCP2 和 ECCP3）实现标准的捕捉和比较模式，以及增强型 PWM 模式。这些在**第 17.0 节 “增强型捕捉 / 比较 /PWM （ECCP）模块”**中讨论。

每个 CCP/ECCP 模块包含一个 16 位寄存器，可用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。为避免混淆，以下所有的 CCP 模块操作描述均针对 CCP4，但同样适用于 CCP5。

在本章中描述的捕捉和比较操作适用于所有标准和增强型 CCP 模块。**第 16.4 节 “PWM 模式”**中描述的 PWM 模式的操作只适用于 CCP4 和 CCP5。

注：	在本节和**第 17.0 节 “增强型捕捉 / 比较 /PWM （ECCP）模块”**中，在提到与特定 CCP 模块相关的寄存器和位名称时，一般会使用 “x” 或 “y” 代替特定的模块编号。因此，“CCPxCON” 可能指 ECCP1、ECCP2、ECCP3、CCP4 或 CCP5 的控制寄存器。

寄存器 16-1： CCPxCON：CCPx 控制寄存器 （CCP4 和 CCP5）

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-6 未用：读为 `0`

bit 5-4 **DCxB1:DCxB0**：CCPx 模块 PWM 占空比 bit 1 和 bit 0

<u>捕捉模式：</u>
未使用。
<u>比较模式：</u>
未使用。
<u>PWM 模式：</u>
这两位是10位PWM占空比的低2位（bit 1和bit 0）。占空比的高8位（DCxB9:DCxB2）在CCPRxL中。

bit 3-0 **CCPxM3:CCPxM0**：CCPx 模块模式选择位

`0000` = 禁止捕捉 / 比较 /PWM （复位 CCPx 模块）
`0001` = 保留
`0010` = 比较模式，匹配时输出电平翻转 （CCPxIF 位置 1）
`0011` = 保留
`0100` = 捕捉模式，每个下降沿
`0101` = 捕捉模式，每个上升沿
`0110` = 捕捉模式，每 4 个上升沿
`0111` = 捕捉模式，每 16 个上升沿
`1000` = 比较模式：初始化 CCPx 引脚为低电平，比较匹配时强制 CCPx 引脚为高电平 （CCPxIF 位置 1）
`1001` = 比较模式：初始化 CCPx 引脚为高电平，比较匹配时强制 CCPx 引脚为低电平 （CCPxIF 位置 1）
`1010` = 比较模式，比较匹配时产生软件中断 （CCPxIF 位置 1， CCPx 引脚反映 I/O 状态）
`1011` = 保留
`11xx` = PWM 模式

16.1 CCP 模块配置

每个捕捉 / 比较 /PWM 模块均与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相对应。数据寄存器由两个 8 位寄存器组成：CCPRxL （低字节）和CCPRxH（高字节）。所有寄存器都是可读写的。

16.1.1 CCP 模块和定时器资源

CCP/ECCP模块根据选定的模式使用Timer1、Timer2、Timer3 或 Timer4。 Timer1 和 Timer3 适用于工作在捕捉或比较模式下的模块，而 Timer2 和 Timer4 适用于工作在 PWM 模式下的模块。

表 16-1： CCP 模式——定时器资源

CCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	Timer1 或 Timer3
PWM	Timer2 或 Timer4

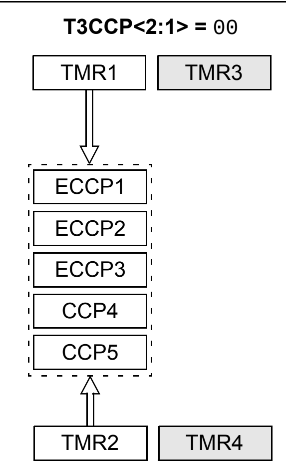
将某个特定的定时器分配给哪个模块是由 T3CON 寄存器 （第 175 页的寄存器 14-1）中的 “Timer-to-CCP”使能位决定的。根据选定的配置，至多同时可以有 4 个定时器有效，具有相同配置 （捕捉 / 比较或 PWM）的模块共用定时器资源。图 16-1 给出了可能的配置情况。

16.1.2 ECCP2 引脚分配

根据器件配置， ECCP2 的引脚分配 （捕捉输入、比较和 PWM 输出）可以更改。CCP2MX 配置位决定哪个引脚将与 ECCP2 复用。默认情况下，ECCP2 引脚被分配给 RC1 （CCP2MX = 1）。如果该配置位被清零，在单片机模式下， ECCP2 将与 RE7 （80 引脚和 100 引脚器件）复用；在扩展单片机模式下，将与 RB3（100 引脚器件）复用。

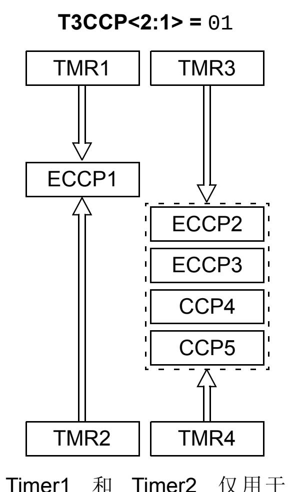
改变 ECCP2 的引脚分配不会自动更改该端口引脚的配置。无论其引脚的分配如何，用户必须始终确保与ECCP2 操作相对应的 TRIS 寄存器配置正确。

图 16-1： CCP 和定时器互连配置



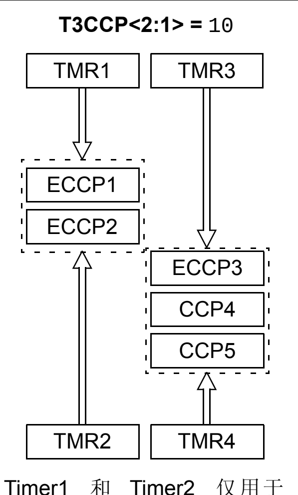
Timer1 用于所有 CCP 模块的所有捕捉和比较操作。Timer2 用于所有 CCP 模块的 PWM 操作。模块可以共用这两个定时器资源中的任何一个作为公共时基。

Timer3 和 Timer4 不可用。



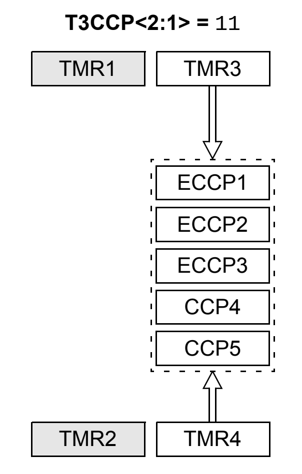
Timer1 和 Timer2 仅用于 ECCP1 的捕捉和比较或 PWM 操作 （取决于选定的模式）。

所有其他模块使用 Timer3 或 Timer4。如果模块处于捕捉 / 比较或 PWM 模式，模块可以共用这两个定时器资源中的任何一个作为公共时基。



Timer1 和 Timer2 仅用于 ECCP1 和 ECCP2 的捕捉和比较或 PWM 操作 （取决于为每个模块选定的模式） 。如果这两个模块都处于捕捉 / 比较或 PWM 模式，则它们可以使用同一个定时器作为公共时基。

其他模块使用 Timer3 或 Timer4。如果模块处于捕捉 / 比较或 PWM 模式，模块可以共用这两个定时器资源中的任何一个作为公共时基。



Timer3 用于所有 CCP 模块的所有捕捉和比较操作。Timer4 用于所有 CCP 模块的 PWM 操作。模块可以共用这两个定时器资源中的任何一个作为公共时基。

Timer1 和 Timer2 不可用。

 超前信息

16.2 捕捉模式

在捕捉模式下，当相应的 CCPx 引脚发生以下事件时，CCPRxH:CCPRxL 寄存器对捕捉 TMR1 或 TMR3 寄存器的 16 位值。事件定义为下列情况之一：

- 每个下降沿
- 每个上升沿
- 每 4 个上升沿
- 每 16 个上升沿

事件由模式选择位 CCPxM3:CCPxM0 （CCPxCON<3:0>）选择。当完成一次捕捉时，中断请求标志位 CCPxIF 置 1 ；它必须用软件清零。如果在读取寄存器 CCPRx 值之前发生了另一次捕捉，那么原来的捕捉值会被新的捕捉值覆盖。

16.2.1 CCP 引脚配置

在捕捉模式下，应通过将相应的 TRIS 方向位置 1 将 CCPx 引脚配置为输入。

注：	如果 RG4/CCP5/P1D 引脚被配置为输出，对该端口的写操作可能产生捕捉条件。

16.2.2 TIMER1/TIMER3 模式选择

用于捕捉功能的定时器（Timer1 和 / 或 Timer3）必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，无法进行捕捉操作。可在 T3CON 寄存器中选择用于每个 CCP 模块的定时器（见第 **16.1.1 节“CCP 模块和定时器资源”**）。

16.2.3 软件中断

当捕捉模式改变时，可能会产生错误的捕捉中断。用户应该保持 CCPxIE 中断允许位清零以避免错误中断。应在工作模式改变后清零中断标志位 CCPxIF。

16.2.4 CCP 预分频器

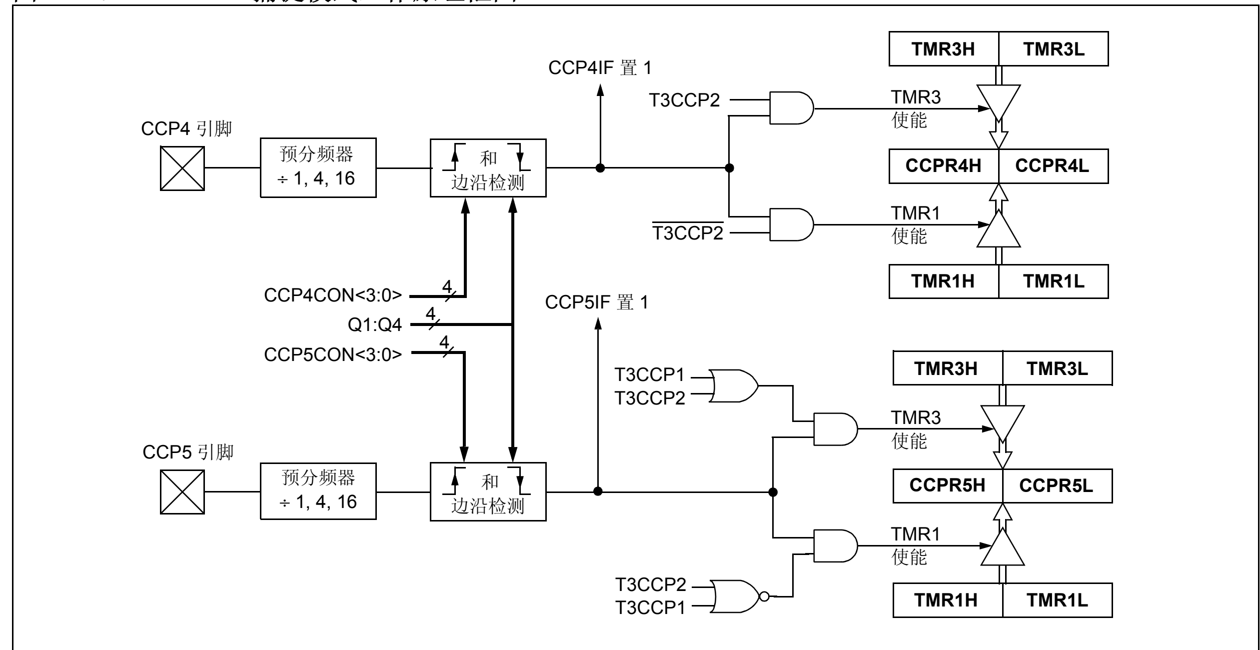
在捕捉模式下有 4 种预分频比设置。它们作为工作模式的一部分由模式选择位（CCPxM3:CCPxM0）选择。只要关闭 CCP 模块或禁止捕捉模式，预分频计数器就会被清零。这意味着任何复位都会将预分频计数器清零。

在两个预分频比之间切换会产生中断。而且，预分频计数器不会被清零；因此第一次捕捉可能来自于一个非零的预分频器。例 16-1 给出了切换捕捉预分频比时建议采用的方法。这个示例使预分频计数器清零且不会产生错误中断。

例 16-1： 改变捕捉预分频比（以 **CCP5** 为例）

```
CLRF   CCP5CON      ; Turn CCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF  CCP5CON      ; Load CCP5CON with
                    ; this value
```

图 16-2： 捕捉模式工作原理框图



16.3 比较模式

在比较模式下，16 位 CCPRx 寄存器的值不断与 TMR1 或 TMR3 寄存器对的值作比较。当两者匹配时， CCPx 引脚将会：

- 驱动为高电平
- 驱动为低电平
- 电平翻转 （高电平变为低电平或低电平变为高电平）
- 保持不变 （即反映 I/O 锁存器的状态）

引脚动作取决于模式选择位 （CCPxM3:CCPxM0）的值。同时，中断标志位 CCPxIF 置 1。

16.3.1 CCP 引脚配置

用户必须通过将相应的 TRIS 位清零，将 CCPx 引脚配置为输出。

> **注：** 清零 CCP5CON 寄存器会将 RG4 比较输出锁存器 （取决于器件配置）强制为默认的低电平。这不是 PORTB 或 PORTC I/O 数据锁存器。

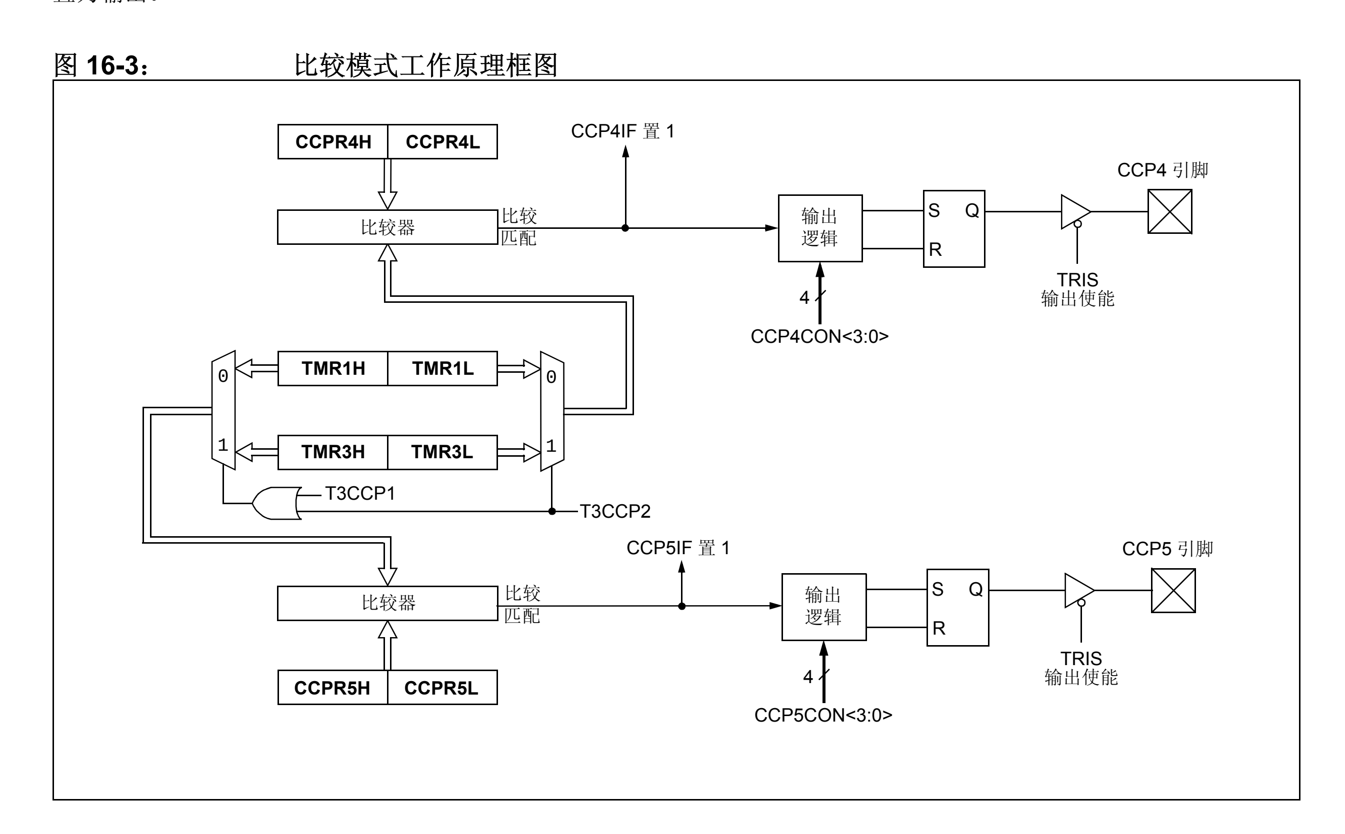
16.3.2 TIMER1/TIMER3 模式选择

如果 CCP 模块使用比较功能，则 Timer1 和 / 或 Timer3 必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，可能无法进行比较操作。

16.3.3 软件中断模式

当选择了“生成软件中断模式”时（CCPxM3:CCPxM0 = `1010`），相应的 CCPx 引脚不受影响。如果中断被允许，将仅产生 CCP 中断并将 CCPxIE 位置 1。

图 16-3： 比较模式工作原理框图



 超前信息

表 16-2：　与捕捉、比较、 TIMER1 和 TIMER3 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
TRISG	TRISG7	TRISG6	TRISG5	TRISG4	TRISG3[1]	TRISG2	TRISG1	TRISG0	61
TMR1L	Timer1 寄存器的低字节								60
TMR1H	Timer1 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	60
TMR3H	Timer3 寄存器的高字节								60
TMR3L	Timer3 寄存器的低字节								60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	60
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								62
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								62
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								63
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								63
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	63
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	63

图注：　— = 未用 （读为 0）， r = 保留。捕捉 / 比较、 Timer1 或 Timer3 不使用阴影单元。

注　1：　该位仅在 80 引脚和 100 引脚器件中可用；在其他器件中未实现且读为 0。

16.4 PWM 模式

在脉宽调制（PWM）模式下， CCPx 引脚会产生高达 10 位分辨率的 PWM 输出信号。由于 CCP4 和 CCP5 引脚与 PORTG 数据锁存器复用，必须清零相应的 TRISG 位才能使 CCP4 或 CCP5 引脚成为输出引脚。

> **注：** 清零 CCP4CON 或 CCP5CON 寄存器会将 RG3 或 RG4 比较输出锁存器（取决于器件配置）强制为默认的低电平。这不是 PORTG 的 I/O 数据锁存器。

图 16-4 给出了 PWM 模式下 CCP 模块的简化框图。

关于如何设置 CCP 模块使之工作于 PWM 模式的详细步骤，请参见**第 16.4.3 节 “设置 PWM 工作模式”**。

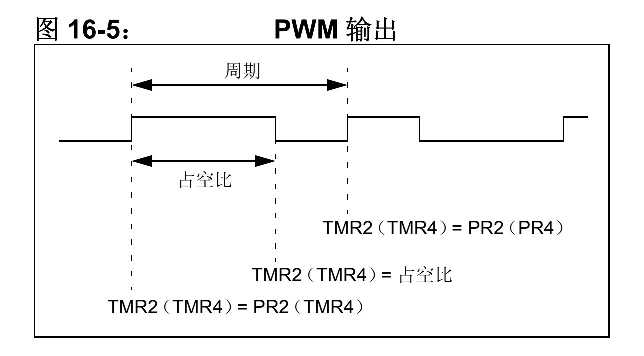
图 16-4： 简化的 PWM 框图

占空比寄存器
9 0
CCPR1L
CCP1CON<5:4>
锁存占空比
CCPR1H (1)
比较器
S Q
R
ECCP1 引脚
复位
TMR2
TMR2 = PR2 匹配
比较器
2 个 LSb 已锁存来自 Q 时钟
PR2
TRIS 输出使能
CCPx 引脚置 1

注 1： 占空比寄存器的两个 LSb 由作为该模块一部分的 2 位锁存器保留。物理上是和 CCPRx 寄存器分离的。

PWM 输出（图 16-5）有一个时基（周期）和一段输出保持为高电平的时间（占空比）。PWM 的频率是周期的倒数（1/ 周期）。

图 16-5： PWM 输出



16.4.1 PWM 周期

可通过写 PR2（PR4）寄存器指定 PWM 周期。 PWM 周期可由公式 16-1 计算：

公式 16-1：

$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1/[PWM 周期]。

当 TMR2（TMR4）中的值与 PR2（PR4）中的值相等时，在下一个递增周期将发生以下 3 个事件：

- TMR2（TMR4）被清零
- CCPx 引脚置 1（例外：如果 PWM 占空比 = 0%，CCPx 引脚将不会置 1）
- PWM 占空比从 CCPRxL 锁存到 CCPRxH

> **注：** 确定PWM频率时不会用到Timer2和Timer4后分频器（见**第 13.0 节 “Timer2 模块”**和**第 15.0 节 “Timer4 模块”**）。后分频器可用不同于PWM输出频率的频率进行数据更新。

16.4.2 PWM 占空比

通过写 CCPRxL 寄存器和 CCPxCON<5:4> 位来指定 PWM 占空比。分辨率最高可达 10 位。CCPRxL 包含占空比的高 8 位而 CCPxCON<5:4> 包含低 2 位。这 10 位值由 CCPRxL:CCPxCON<5:4> 表示。公式 16-2 用于计算 PWM 的占空比时间。

公式 16-2：

$$\text{PWM 占空比} = (CCPRxL:CCPxCON<5:4>) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可以在任何时候写入 CCPRxL 和 CCPxCON<5:4>，但是在 PR2（PR4）和 TMR2（TMR4）发生匹配（即周期结束）前占空比值不会被锁存到 CCPRxH 中。在 PWM 模式下， CCPRxH 是只读寄存器。

CCPRxH 寄存器和一个 2 位的内部锁存器用于给 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，它可以避免在 PWM 操作中产生毛刺。

当 CCPRxH 和 2 位锁存值与 TMR2（TMR4）（以及内部 2 位 Q 时钟或 TMR2（TMR4）预分频值的 2 位）匹配时， CCPx 引脚被清零。

在给定 PWM 频率的情况下，最大的 PWM 分辨率（位）由公式 16-3 给出：

公式 16-3：

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ 位}$$

注：	如果 PWM 占空比的值大于 PWM 周期，则 CCPx 引脚将不会被清零。

16.4.3 设置 PWM 工作模式

当配置 CCP 模块使之工作于 PWM 模式时，应遵循以下步骤：

1. 通过写 PR2 （PR4）寄存器设置 PWM 周期。
2. 通过写 CCPRxL 寄存器和 CCPxCON<5:4> 位设置 PWM 占空比。
3. 通过清零相应的 TRIS 位将 CCPx 引脚设为输出引脚。
4. 设置 TMR2 （TMR4）预分频值并随后通过写 T2CON （T4CON）使能 Timer2 （Timer4）。
5. 配置 CCPx 模块使之工作于 PWM 模式。

表 16-3： 40 MHz 时 PWM 频率和分辨率示例

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频值 （1、 4 和 16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最大分辨率 （位）	10	10	10	8	7	6.58

表 16-4： 与 PWM、 TIMER2 和 TIMER4 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
TRISG	TRISG7	TRISG6	TRISG5	TRISG4	TRISG3[1]	TRISG2	TRISG1	TRISG0	61
TMR2	Timer2 寄存器								60
PR2	Timer2 周期寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
TMR4	Timer4 寄存器								62
PR4	Timer4 周期寄存器								62
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	62
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								62
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								62
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								63
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								63
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	63
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	63

图注： — = 未用 （读为 0）。 PWM、 Timer2 或 Timer4 不使用阴影单元。

注 1： 该位仅在 80 引脚和 100 引脚器件中可用；在其他器件中未实现且读为 0。

 超前信息

17.0 增强型捕捉 / 比较 /PWM （ECCP）模块

在 PIC18F97J60 系列器件中，有 3 个 CCP 模块实现为具有增强型 PWM 功能的标准 CCP 模块。这些功能包括 2 或 4 路输出通道、用户可选极性、死区控制和自动关闭与重启。**第 17.4 节 “增强型 PWM 模式”**将详细讨论增强功能。ECCP 模块的捕捉、比较和单输出 PWM 功能与标准 CCP 模块的相同。

增强型 CCP 模块的控制寄存器如寄存器 17-1 所示。它与 CCP4CON/CCP5CON 寄存器的不同之处在于，它有两个最高有效位用来控制 PWM 功能。

除了通过增强型 CCPxCON 寄存器提供的扩展模式外，每个 ECCP 模块还有两个与增强型 PWM 操作和自动关闭功能相关的寄存器。它们是：

- ECCPxDEL （死区延时）
- ECCPxAS （自动关闭配置）

寄存器 17-1： CCPxCON：增强型 CCPx 控制寄存器 （ECCP1/ECCP2/ECCP3）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-6 **PxM1:PxM0**：增强型 PWM 输出配置位
如果 CCPxM3:CCPxM2 = 00、 01 和 10：
xx = PxA 配置为捕捉 / 比较输入 / 输出； PxB、 PxC 和 PxD 配置为端口引脚
如果 CCPxM3:CCPxM2 = 11：
00 = 单输出：PxA 被调制； PxB、 PxC 和 PxD 配置为端口引脚
01 = 全桥正向输出：P1D 被调制； P1A 有效； P1B 和 P1C 无效
10 = 半桥输出：P1A 和 P1B 被调制，带有死区控制； P1C 和 P1D 配置为端口引脚
11 = 全桥反向输出：P1B 被调制； P1C 有效； P1A 和 P1D 无效

bit 5-4 **DCxB1:DCxB0**：ECCPx 模块 PWM 占空比 bit 1 和 bit 0
捕捉模式：
未使用。
比较模式：
未使用。
PWM 模式：
这些位是 10 位 PWM 占空比的低 2 位。占空比的高 8 位在 CCPRxL 中。

bit 3-0 **CCPxM3:CCPxM0**：ECCPx 模块模式选择位
0000 = 禁止捕捉 / 比较 /PWM （复位 ECCPx 模块）
0001 = 保留
0010 = 比较模式：匹配时翻转输出
0011 = 捕捉模式
0100 = 捕捉模式，每个下降沿
0101 = 捕捉模式，每个上升沿
0110 = 捕捉模式，每 4 个上升沿
0111 = 捕捉模式，每 16 个上升沿
1000 = 比较模式：初始化 ECCPx 引脚为低电平；比较匹配时输出置 1 （CCPxIF 置 1）
1001 = 比较模式：初始化 ECCPx 引脚为高电平；比较匹配时输出清零 （CCPxIF 置 1）
1010 = 比较模式：仅产生软件中断； ECCPx 引脚回复到 I/O 状态
1011 = 比较模式：触发特殊事件（ECCPx 复位 TMR1 或 TMR3， CCPxIF 位置 1，如果使能了 A/D 模块， ECCP2 触发器还将启动 A/D 转换）[1]
1100 = PWM 模式：PxA 和 PxC 高电平有效； PxB 和 PxD 高电平有效
1101 = PWM 模式：PxA 和 PxC 高电平有效； PxB 和 PxD 低电平有效
1110 = PWM 模式：PxA 和 PxC 低电平有效； PxB 和 PxD 高电平有效
1111 = PWM 模式：PxA 和 PxC 低电平有效； PxB 和 PxD 低电平有效

注 1： 只在 ECCP1 和 ECCP2 上实现；对于 ECCP3，它和 1010 相同。

17.1 ECCP 输出和配置

取决于所选定的工作模式，每个增强型 CCP 模块最多有四路 PWM 输出。这些指定为 PxA 到 PxD 的输出，可以与不同的 I/O 引脚复用。某些 ECCPx 引脚分配是不变的，其他引脚分配则根据器件配置而变化。对于那些可变引脚，控制位是：

- CCP2MX 配置位（仅限 80 引脚和 100 引脚器件）
- ECCPMX 配置位（仅限 80 引脚和 100 引脚器件）
- 设置程序存储器操作模式的 EMB 配置位 （仅限 100 引脚器件）

表 17-1、表 17-2 和表 17-3 中总结了增强型 CCP 模块的引脚分配。若要将 I/O 引脚配置为 PWM 输出，必须通过设置PxMx和CCPxMx位（分别是CCPxCON<7:6> 和 <3:0>）来选择适当的 PWM 模式。相应端口引脚的相应 TRIS 方向位也必须设置为输出。

17.1.1 ECCP1/ECCP3 输出和程序存储器模式

在 100 引脚器件中，使用扩展单片机模式会对增强型 PWM 模式下的 ECCP1 和 ECCP3 引脚产生间接影响。默认情况下， PWM 输出 P1B/P1C 和 P3B/P3C 将与 PORTE 引脚以及外部存储器总线的高位字节复用。总线在扩展单片机模式下工作时，它将改写增强型CCP输出，使它们不可用。因此，器件处于扩展单片机模式和默认引脚配置时，ECCP1 和 ECCP3 只能在兼容的（单输出） PWM 模式下使用。

该配置有一种例外情况，即为外部总线选择 12 位地址宽度时 （EMB1:EMB0 配置位 = `10`）。在这种情况下，即使外部总线有效，PORTE的高位引脚仍用作数字I/O。P1B/P1C 和 P3B/P3C 仍可用作增强型 PWM 输出。

如果在扩展单片机模式中应用要求使用更多的 PWM 输出，可以将 P1B/P1C 和 P3B/P3C 输出重新分配到 PORTH 的高位。这是通过将 ECCPMX 配置位清零实现的。

17.1.2 ECCP2 输出和程序存储器模式

对于 100 引脚器件，器件的程序存储器模式（**第 5.1.3 节 “PIC18F9XJ60/9XJ65 程序存储器模式”**）也会影响该模块的引脚复用。

ECCP2 输入 / 输出（ECCP2/P2A）可以与 3 个引脚之一复用。所有器件的默认分配（CCP2MX 配置位置 1）是 RC1。清零 CCP2MX 会将 ECCP2/P2A 重新分配到 RE7。

100 引脚器件有另外一个选项。这些器件在单片机模式下工作时，上述的复用选项仍适用。在扩展单片机模式下，清零 CCP2MX 会将 ECCP2/P2A 重新分配到 RB3。

17.1.3 CCP4 和 CCP5 与 ECCP1 和 ECCP3 一起使用

只有 ECCP2 模块有 4 个专用输出引脚可用。假如不需要那些引脚上的 I/O 端口或其他复用功能，则它们在使用时就不会牵扯到任何其他 CCP 模块。

另一方面，ECCP1 和 ECCP3 只有 3 个专用输出引脚：ECCPx/PxA、PxB 和 PxC。当这些模块配置为四 PWM 模式时，通常用于CCP4或CCP5的引脚就变为ECCP3 或 ECCP1 的 PxD 输出引脚。CCP4 和 CCP5 模块仍正常工作，但其输出会被改写。

17.1.4 ECCP 模块和定时器资源

与标准的 CCP 模块一样， ECCP 模块根据选定的模式使用 Timer1、Timer2、Timer3 或 Timer4。该模块在捕捉或比较模式下使用 Timer1 和 Timer3，而在 PWM 模式下使用 Timer2 和 Timer4。**第 16.1.1 节 “CCP 模块和定时器资源”**给出了定时器资源的更多详细信息。

表 17-1： ECCP1 的引脚配置

ECCP 模式	CCP1CON 配置	RC2	RD0 或 RE6(1)	RE5	RG4	RH7(2)	RH6(2)
64 引脚器件； 80 引脚器件， ECCPMX = 1 ； 100 引脚器件， ECCPMX = 1，单片机模式或扩展单片机模式， 12 位地址宽度：							
兼容的 CCP	00xx 11xx	ECCP1	RD0/RE6	RE5	RG4/CCP5	RH7/AN15	RH6/AN14
双 PWM	10xx 11xx	P1A	P1B	RE5	RG4/CCP5	RH7/AN15	RH6/AN14
四 PWM	x1xx 11xx	P1A	P1B	P1C	P1D	RH7/AN15	RH6/AN14
80 引脚器件， ECCPMX = 0 ； 100 引脚器件， ECCPMX = 0，所有程序存储器模式：							
兼容的 CCP	00xx 11xx	ECCP1	RD0/RE6	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
双 PWM	10xx 11xx	P1A	RD0/RE6	RE5/AD13	RG4/CCP5	P1B	RH6/AN14
四 PWM(3)	x1xx 11xx	P1A	RD0/RE6	RE5/AD13	P1D	P1B	P1C
100 引脚器件， ECCPMX = 1，扩展单片机模式， 16 位或 20 位地址宽度：							
兼容的 CCP	00xx 11xx	ECCP1	RD0/RE6	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14

图注： x = 无关位， N/A = 不可用。阴影单元表示在给定模式下 ECCP1 不使用的引脚分配。

注 1： P1B 与 64 引脚器件上的 RD0、 80 引脚和 100 引脚器件上的 RE6 复用。

2： 64 引脚器件上不提供这些引脚选项。

3： ECCP1 处于四 PWM 模式时， CCP5 引脚的输出被 P1D 改写；其他模式下， CCP5 完全正常工作。

表 17-2： ECCP2 的引脚配置

ECCP 模式	CCP2CON 配置	RB3	RC1	RE7	RE2	RE1	RE0
所有器件， CCP2MX = 1，所有程序存储器模式：							
兼容的 CCP	00xx 11xx	RB3/INT3	ECCP2	RE7	RE2	RE1	RE0
双 PWM	10xx 11xx	RB3/INT3	P2A	RE7	P2B	RE1	RE0
四 PWM	x1xx 11xx	RB3/INT3	P2A	RE7	P2B	P2C	P2D
80 引脚和 100 引脚器件， CCP2MX = 0，单片机模式：							
兼容的 CCP	00xx 11xx	RB3/INT3	RC1/T1OS1	ECCP2	RE2	RE1	RE0
双 PWM	10xx 11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	RE1	RE0
四 PWM	x1xx 11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	P2C	P2D
100 引脚器件， CCP2MX = 0，扩展单片机模式：							
兼容的 CCP	00xx 11xx	ECCP2	RC1/T1OS1	RE7/AD15	RE2/$\overline{CS}$	RE1/$\overline{WR}$	RE0/$\overline{RD}$
双 PWM	10xx 11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	RE1/$\overline{WR}$	RE0/$\overline{RD}$
四 PWM	x1xx 11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	P2C	P2D

图注： x = 无关位。阴影单元表示在给定模式下 ECCP2 不使用的引脚分配。

表 17-3： ECCP3 的引脚配置

ECCP 模式	CCP3CON 配置	RD1 或 RG0(1)	RE4	RE3	RD2 或 RG3(1)	RH5(2)	RH4(2)
64 引脚器件； 80 引脚器件， ECCPMX = 1 ； 100 引脚器件， ECCPMX = 1，单片机模式：							
兼容的 CCP	00xx 11xx	ECCP3	RE4	RE3	RD2/RG3	RH5/AN13	RH4/AN12
双 PWM	10xx 11xx	P3A	P3B	RE3	RD2/RG3	RH5/AN13	RH4/AN12
四 PWM	x1xx 11xx	P3A	P3B	P3C	P3D	RH5/AN13	RH4/AN12
80 引脚器件， ECCPMX = 0 ； 100 引脚器件， ECCPMX = 0，所有程序存储器模式：							
兼容的 CCP	00xx 11xx	ECCP3	RE6/AD14	RE5/AD13	RD2/RG3	RH5/AN13	RH4/AN12
双 PWM	10xx 11xx	P3A	RE6/AD14	RE5/AD13	RD2/RG3	P3B	RH4/AN12
四 PWM(3)	x1xx 11xx	P3A	RE6/AD14	RE5/AD13	P3D	P3B	P3C
100 引脚器件， ECCPMX = 1，扩展单片机模式， 12 位地址宽度：							
兼容的 CCP	00xx 11xx	ECCP3	RE4/AD12	RE3/AD11	RD2/RG3	RH5/AN13	RH4/AN12
双 PWM	10xx 11xx	P3A	P3B	RE3/AD11	RD2/RG3	RH5/AN13	RH4/AN12
100 引脚器件， ECCPMX = 1，扩展单片机模式， 16 位或 20 位地址宽度：							
兼容的 CCP	00xx 11xx	ECCP3	RE6/AD14	RE5/AD13	RD2/RG3	RH5/AN13	RH4/AN12

图注： x = 无关位， N/A = 不可用。阴影单元表示在给定模式下 ECCP3 不使用的引脚分配。

注 1： ECCP3/P3A 和 CCP4/P3D 与 64 引脚器件上的 RD1 和 RD2、 80 引脚和 100 引脚器件上的 RG0 和 RG3 复用。

2： 64 引脚器件上不提供这些引脚选项。

3： ECCP3 处于四 PWM 模式时， CCP4 引脚的输出被 P3D 改写；其他模式下， CCP4 完全正常工作。

17.2 捕捉和比较模式

除了下面讨论的特殊事件触发器的操作， ECCP 模块的捕捉和比较模式与 CCP4 的操作是相同的。这些已在**第 16.2 节 "捕捉模式"**和**第 16.3 节 "比较模式"**中有详细讨论。

17.2.1 特殊事件触发器

ECCP1 和 ECCP2 包含内部硬件触发器，在比较模式下， CCPRx 寄存器对与选定的定时器匹配时，它会产生输出。其输出又可用于启动一项操作。该模式可通过将 CCPxCON<3:0> 设置为 1011 而选中。

ECCP1 或 ECCP2 的特殊事件触发信号会复位 TMR1 或 TMR3 寄存器对，具体复位哪一对寄存器，视当前选定的定时器资源而定。这使得 CCPRx 寄存器实际上成为 Timer1 或 Timer3 的 16 位可编程周期寄存器。此外，如果 A/D 模块被使能， ECCP2 特殊事件触发信号还会启动一次 A/D 转换。

在 ECCP3、CCP4 或 CCP5 上未实现特殊事件触发器。为这些模块选择特殊事件触发器模式与选择 "软件中断比较" 模式 （CCPxM3:CCPxM0 = 1010）具有相同效果。

注：	ECCP2的特殊事件触发信号不会将Timer1 或 Timer3 中断标志位置 1。

17.3 标准 PWM 模式

当配置为单输出模式时，ECCP 模块的功能与 PWM 模式下的标准 CCP 模块相同，如**第 16.4 节 "PWM 模式"**中所述。有时也称为 "兼容的 CCP" 模式，如表 17-1 到 17-3 中所示。

注：	当设置单输出 PWM操作时，用户可自由使用**第 16.4.3 节 "设置 PWM 工作模式"**或**第 17.4.9 节 "设置 PWM 工作模式"**中所述的任一过程。后者更为通用，并可用于单输出或多输出 PWM。

17.4 增强型 PWM 模式

增强型 PWM 模式为更大范围的控制应用提供了更多的 PWM 输出选项。该模块是标准 CCP 模块的后向兼容版本，可提供最多四路输出，指定为 PxA 到 PxD。用户还可以选择信号的极性（高电平有效或低电平有效）。该模块的输出模式和极性可通过设置CCPxCON寄存器的 PxM1:PxM0 和 CCPxM3:CCPxM0 位（分别是 CCPxCON<7:6> 和 CCPxCON<3:0>）配置。

为清楚起见，在本节中针对 ECCP1 和 TMR2 模块概述了增强型PWM模式的操作。控制寄存器名称用ECCP1的代表。所有 3 个增强型模块以及 2 个定时器都可以互换使用，功能相同。可通过选择 T3CON 中正确的位，为 PWM 操作选择 TMR2 或 TMR4。

图 17-1 给出了 PWM 工作原理的简化框图。所有控制寄存器都是双重缓冲的，且在新的 PWM 周期（Timer2 复位时的周期边界）开始时装入，以防止任何输出出现毛刺。ECCP1 死区延时寄存器 ECCP1DEL 是个例外，它在占空比边界或者周期边界装入（取决于哪一个先出现）。由于缓冲作用，模块会一直等到指定的定时器复位，而不是立即开始。这意味着增强型 PWM 波形与标准 PWM 波形并不完全一致，而是偏移一个指令周期（4 TOSC）。

像以前一样，用户必须手动将相应的 TRIS 位设置为输出。

17.4.1 PWM 周期

可通过写 PR2 寄存器指定 PWM 周期。PWM 周期可由以下公式计算：

公式 17-1：

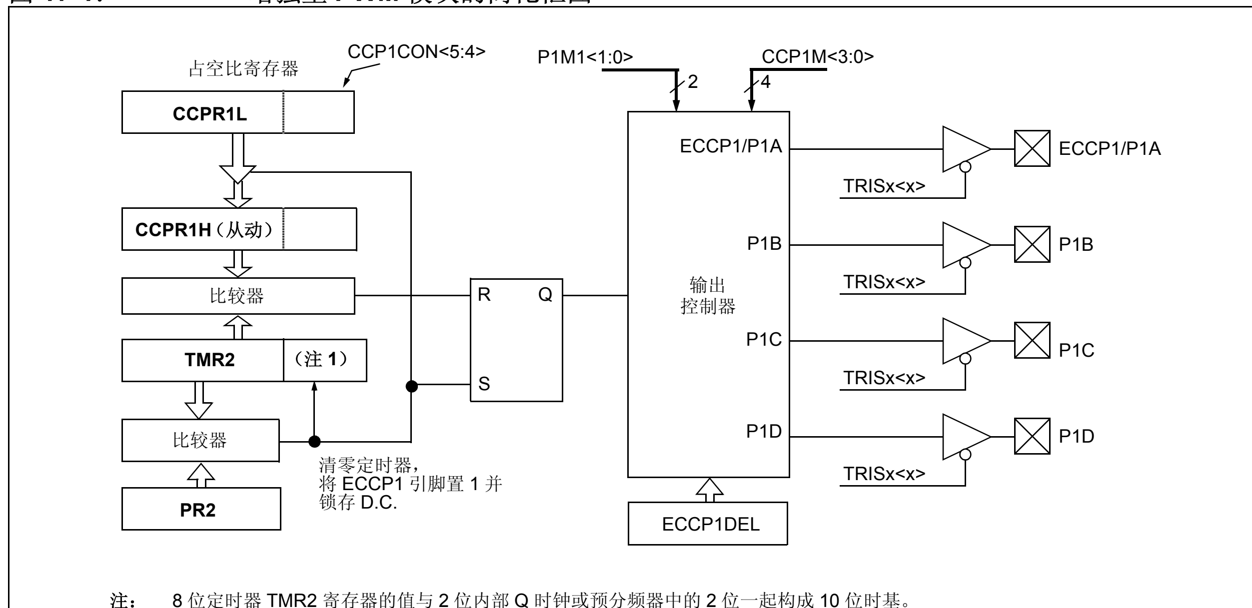
$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1/[PWM 周期]。当 TMR2 中的值与 PR2 中的值相等时，在下一个递增周期将发生以下 3 个事件：

- TMR2 被清零
- ECCP1 引脚置 1 （如果 PWM 占空比 = 0%，ECCP1 引脚不会被置 1）
- PWM 占空比从 CCPR1L 复制到 CCPR1H

注：	在确定 PWM 频率时不会用到 Timer2 后分频器（见**第 13.0 节 “Timer2 模块”**）。后分频器可用于获得与PWM输出不同频率的伺服更新速率。

图 17-1： 增强型 PWM 模块的简化框图



17.4.2 PWM 占空比

通过写 CCPR1L 寄存器和 CCP1CON<5:4> 位来指定 PWM 占空比。分辨率最高可达 10 位。CCPR1L 包含高 8 位而 CCP1CON<5:4> 包含低 2 位。这 10 位值由 CCPRxL:CCPxCON<5:4> 表示。PWM 占空比可由以下公式计算：

公式 17-2：

$$\text{PWM 占空比} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可以在任何时候写入 CCPR1L 和 CCP1CON<5:4>，但是在 PR2 和 TMR2 发生匹配 （即周期结束）前占空比值不会被复制到 CCPR1H 中。在 PWM 模式下，CCPR1H 是只读寄存器。

CCPRxH 寄存器和一个 2 位的内部锁存器用于给 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，它可以避免在 PWM 操作中产生毛刺。当 CCPR1H 和 2 位锁存值与 TMR2，及内部 2 位 Q 时钟或 2 位 TMR2 预分频器值相匹配时， ECCP1 引脚被清零。在给定 PWM 频率的情况下，最大的 PWM 分辨率 （位）由以下公式给出：

公式 17-3：

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ 位}$$

> **注：** 如果 PWM 占空比的值大于 PWM 周期，则 ECCP1 引脚将不会被清零。

17.4.3 PWM 输出配置

CCP1CON 寄存器的 P1M1:P1M0 位可用于设置以下 4 种配置之一：

- 单输出
- 半桥输出
- 全桥输出，正向模式
- 全桥输出，反向模式

单输出模式也就是在**第 17.4 节 “增强型 PWM 模式”**中讨论的标准 PWM 模式。半桥和全桥输出模式在接下来的章节中详细讨论。

图 17-2 总结了所有配置输出的对应关系。

表 17-4： 40 MHz 时 PWM 频率和分辨率示例

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频值（1、4 和 16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最大分辨率 （位）	10	10	10	8	7	6.58

图 17-2： PWM 输出关系 （高电平有效状态）

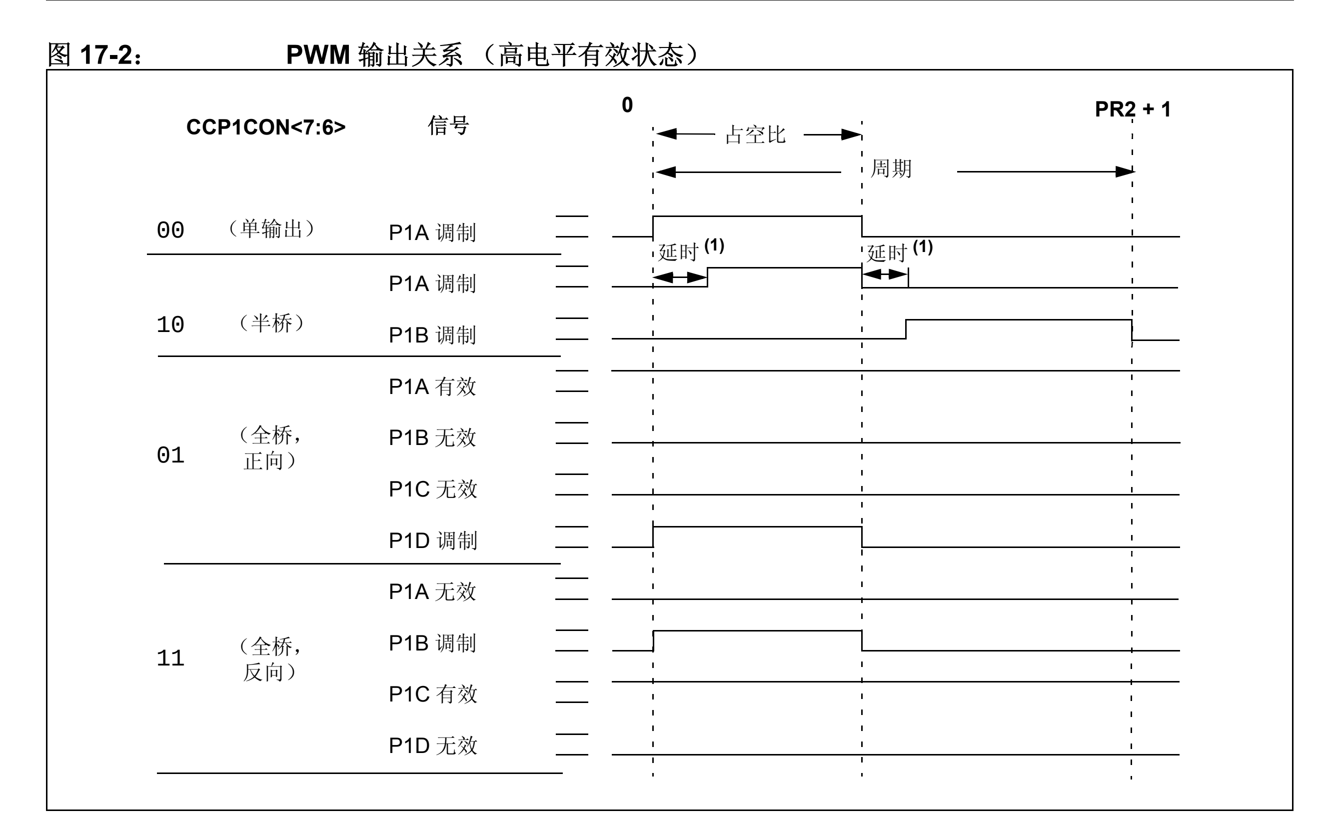
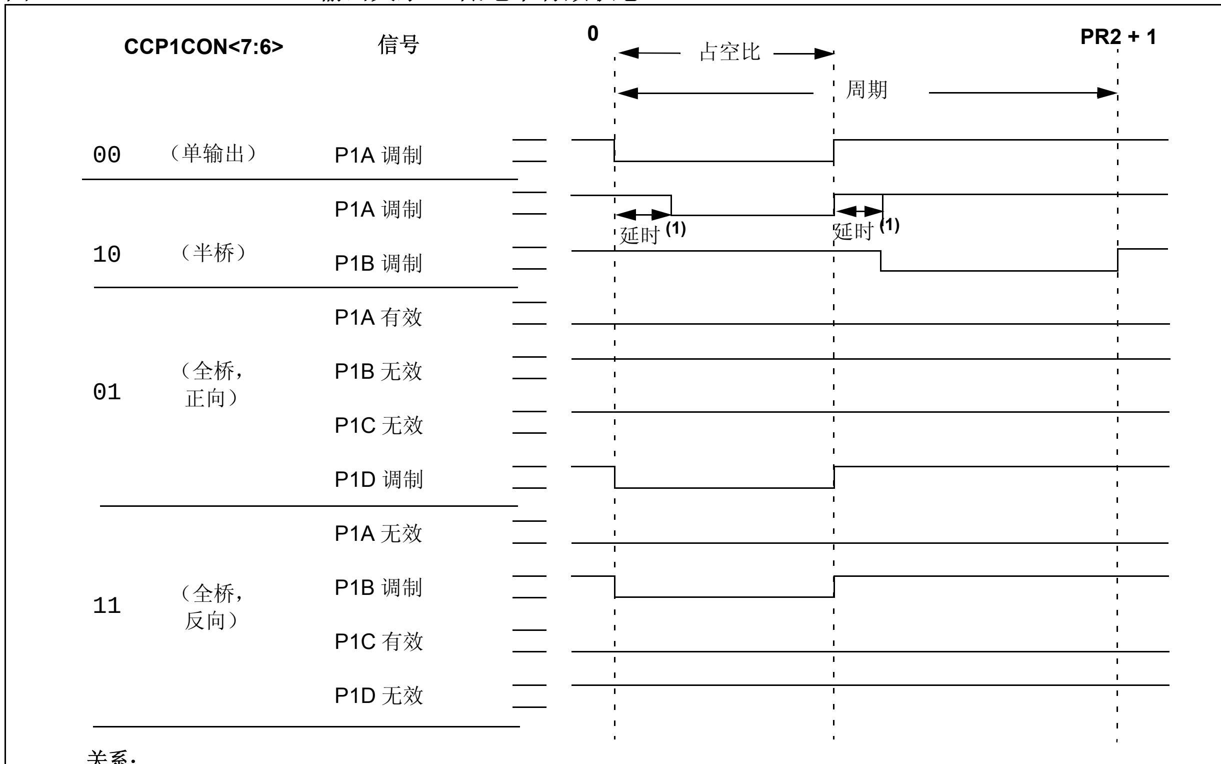




图 17-3： PWM 输出关系 （低电平有效状态）



关系：

- 周期 = 4 * Tosc * (PR2 + 1) * (TMR2 预分频值)
- 占空比 = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 预分频值)
- 延时 = 4 * Tosc * (ECCP1DEL<6:0>)

注 1： 使用 ECCP1DEL 寄存器对死区延时编程 （见**第 17.4.6 节 “可编程死区延时”**）。

17.4.4 半桥模式

在半桥输出模式下，有两个引脚用作输出驱动推挽式负载。P1A 引脚输出 PWM 输出信号，P1B 引脚输出互补的 PWM 输出信号（图 17-4）。这种模式可用于半桥应用（如图 17-5 所示），或者用于全桥应用，这种情况下使用两个 PWM 信号调制 4 个功率开关。

在半桥输出模式下，可编程死区延时可用来防止半桥功率器件中流过直通（Shoot-through）电流。PxDC6:PxDC0 位的值对应在输出被驱动为有效之前的指令周期数。如果这个值比占空比大，则在整个周期中相应的输出保持为无效。关于死区延时操作的详细信息，请参见**第 17.4.6 节 “可编程死区延时”**。

由于 P1A 和 P1B 输出与 PORTC<2> 和 PORTE<6> 数据锁存是复用的， TRISC<2> 和 TRISE<6> 位必须清零，将 P1A 和 P1B 配置为输出。

图 17-4： 半桥 PWM 输出

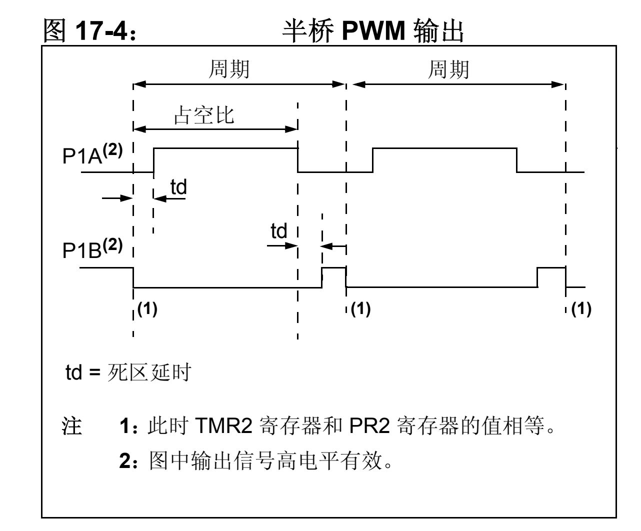


图 17-5： 半桥输出模式应用示例

标准半桥电路（“推挽式”）

PIC18F97J60
P1A
P1B
FET 驱动器
FET 驱动器
V+
V-
负载
+ V -
+ V -

驱动全桥电路的半桥输出

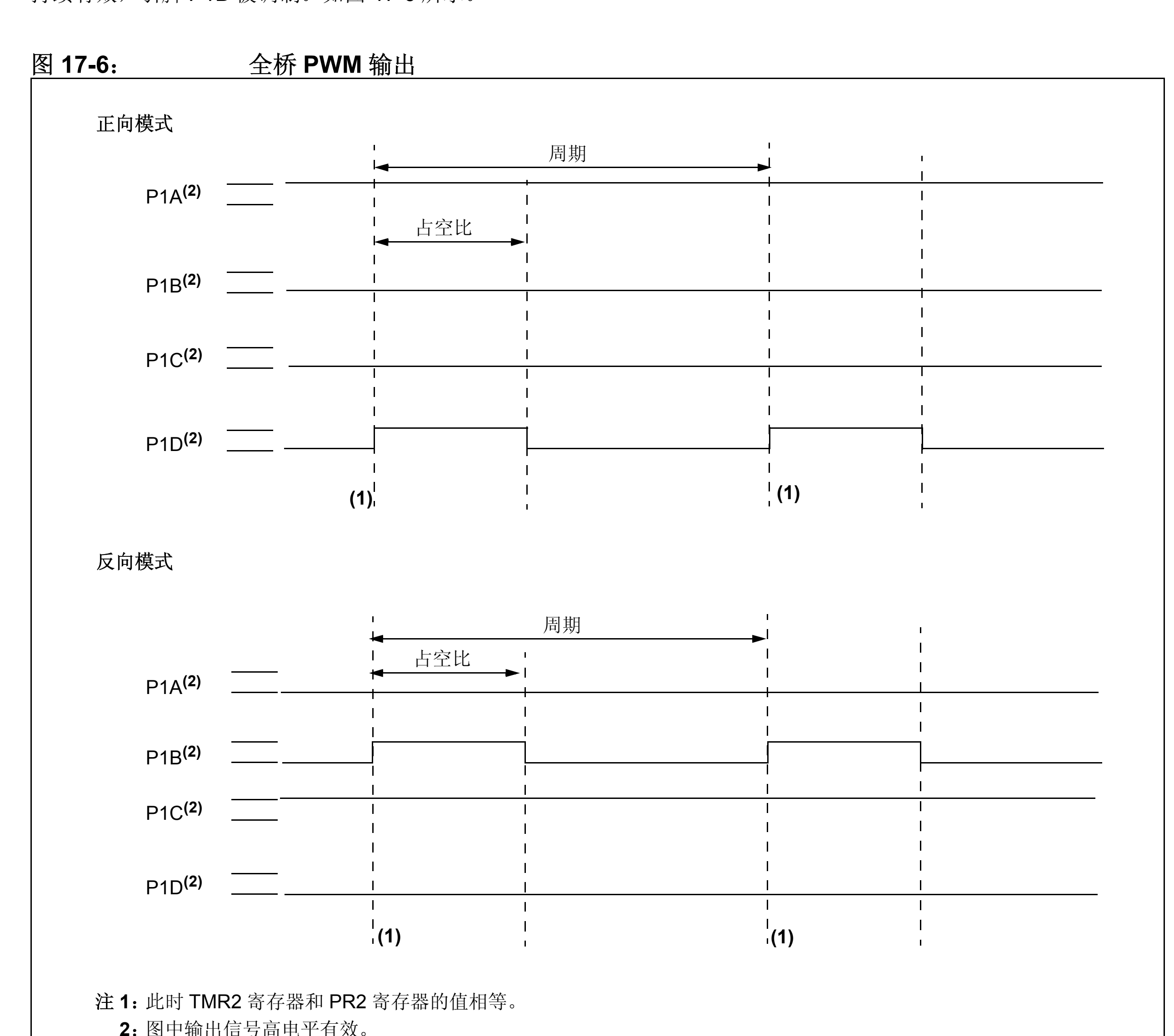
PIC18F97J60
P1A
P1B
FET 驱动器
FET 驱动器
FET 驱动器
FET 驱动器
V+
V-
负载

17.4.5 全桥模式

在全桥输出模式下，四个引脚都用作输出；但是，同一时间只有两个输出同时有效。在正向模式下，引脚 P1A 持续有效，引脚 P1D 被调制。在反向模式下，引脚 P1C 持续有效，引脚 P1B 被调制。如图 17-6 所示。

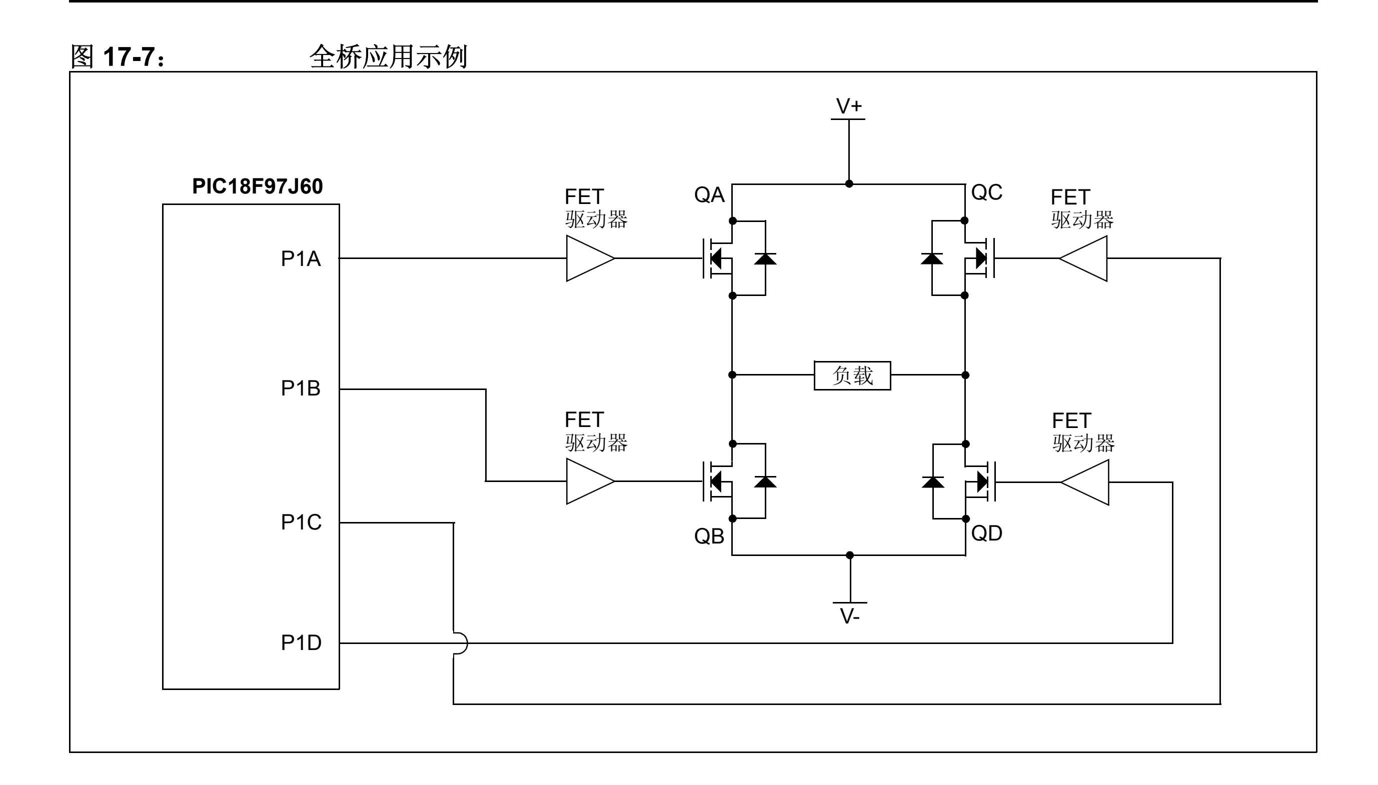
P1A、P1B、P1C 和 P1D 输出与表 17-1 和表 17-3 中列出的端口引脚的数据锁存器复用。相应的 TRIS 位必须清零，以便将P1A、P1B、P1C和P1D引脚配置为输出。

图 17-6： 全桥 PWM 输出



注 1：此时 TMR2 寄存器和 PR2 寄存器的值相等。
2：图中输出信号高电平有效。

图 17-7： 全桥应用示例



17.4.5.1 全桥模式中的方向改变

在全桥输出模式下，CCP1CON 寄存器中的 P1M1 位允许用户控制正 / 反方向。当应用软件改变这个方向控制位时，模块将在下一个 PWM 周期改用新的方向。

在当前 PWM 周期结束之前，调制输出（P1B 和 P1D）进入无效状态，而未调制输出 （P1A 和 P1C）切换到以相反的方向驱动。这发生在下一个 PWM 周期开始之前的（4 Tosc * (Timer2 预分频比)）时间间隔内。Timer2 预分频比可以是 1、 4 或 16，这取决于 T2CKPS 位（T2CON<1:0>）的值。从切换未调制输出驱动方向到下一个周期开始这一段时间内，调制输出 （P1B 和 P1D）保持无效。这种关系如图 17-8 所示。

注意在全桥输出模式下， ECCP1 模块不提供任何死区延时。通常，因为始终只有一个输出被调制，所以不需要死区延时。然而，有一种情况将需要死区延时。这一情况发生在以下两个条件同时满足时：

1. 当输出的占空比达到或者接近 100% 时， PWM 输出方向改变。
2. 功率开关（包括功率器件和驱动电路）的关断时间比开通的时间要长。

在图 17-9 所示的示例中，在占空比接近 100% 时，PWM 方向从正向改变到反向。在时间 t1，输出 P1A 和 P1D 变为无效，而输出 P1C 变为有效。在这个示例中，因为功率器件的关断时间比开通时间要长，在 “t” 时间内，功率器件 QC 和 QD 中可能流过直通电流（见图 17-7）。当 PWM 方向从反向改变到正向时，功率器件 QA 和 QB 也将出现相同的现象。

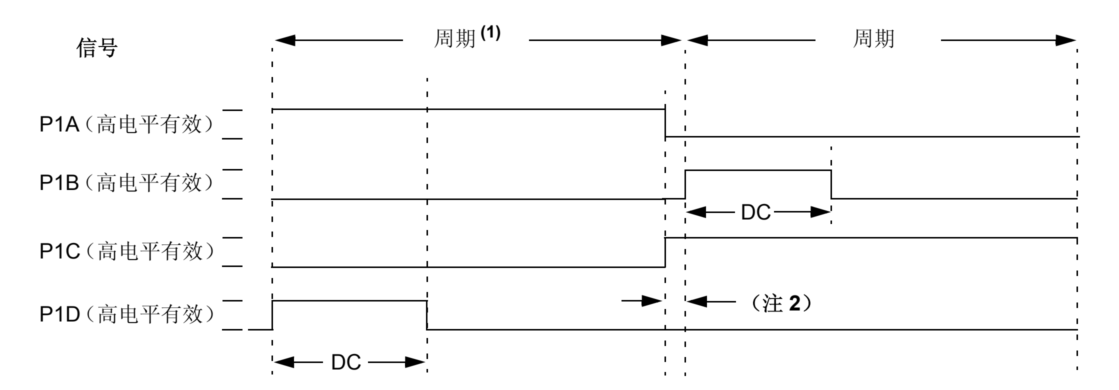
如果应用中需要在高占空比时改变 PWM 方向，必须满足以下要求之一：

1. 在改变方向之前的一个 PWM 周期降低 PWM 占空比。
2. 使用开关驱动电路，使开关管关断的时间比导通的时间短。

也可能存在其他避免直通电流的方案。



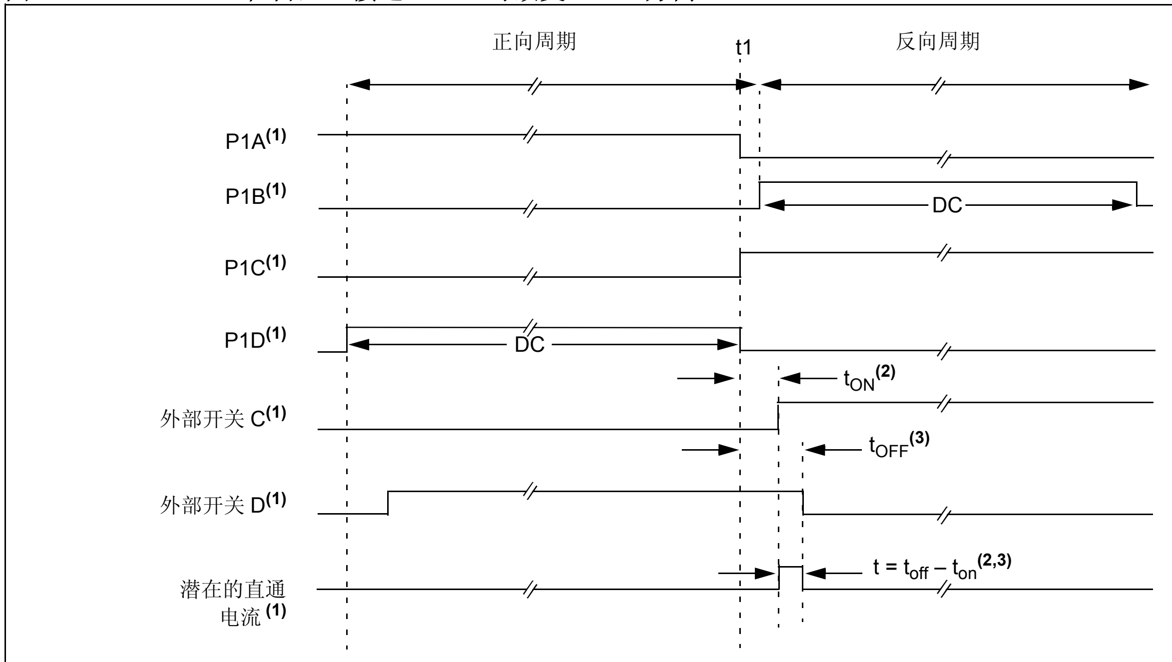
图 17-8： PWM 方向改变



注 **1：** 可在 PWM 周期的任意时刻，写 ECCP1 控制寄存器的方向位 （CCP1CON<7>）。

2： 当改变方向时， P1A 和 P1C 信号在当前 PWM 周期结束之前的 4 TOSC、 16 TOSC 或者 64 TOSC 时间间隔内切换方向，时间长短取决于 Timer2 预分频值。此时被调制的 P1B 和 P1D 信号是无效的。

图 17-9： 在占空比接近 100% 时改变 PWM 方向



注 **1：** 图中所有信号均为高电平有效。

2： t_{ON} 是功率开关 QC 及其驱动电路的导通延时。

3： t_{OFF} 是功率开关 QD 及其驱动电路的关断延时。

17.4.6 可编程死区延时

在所有功率开关管都以 PWM 频率调制的半桥应用中，功率开关管关断通常需要更多的时间。如果上下两个功率开关管在同一时间开关 （一个开通，另一个关断），那么在一段很短的时间里，两个开关管可能同时导通，直到一个开关管完全关断为止。在这短暂的时间中，两个功率开关管中可能流过较高的电流 （*直通电流*），将逆变桥的电源与地短路。为避免开关过程中可能会出现的破坏性直通电流，通常需要延迟功率开关管的导通，保证在另一个开关管完全关断之后，再导通相应的功率开关管。

在半桥输出模式下，可采用数字可编程死区延时来避免出现损坏逆变桥功率开关管的直通电流。在信号从无效状态切换到有效状态时增加延时。请参见图 17-4。ECCPxDEL 寄存器 （寄存器 17-2）的低 7 位以单片机指令周期 （TCY 或 4 TOSC）为单位设置延时。

17.4.7 增强型 PWM 自动关闭

当 ECCP1 编程设置为任一种增强型 PWM 模式时，有效输出引脚可配置为自动关闭。当关闭事件发生时，可立即把增强型 PWM 输出引脚置为定义的关闭状态。

关闭事件可能由两个比较器模块或 FLT0 引脚 （或这三个来源的任意组合）引起。比较器可用来监控桥电路中与被监控电流成比例的电压输入。如果电压超过门限值，比较器将切换状态并触发关闭。或者，也可以由 FLT0 引脚上的低电平数字信号来触发关闭。不选择任何自动关闭源可以禁止自动关闭功能。使用的自动关闭源由 ECCP1AS2:ECCP1AS0 位 （ECCP1AS 寄存器的 <6:4> 位）选择。

当关闭发生时，输出引脚被异步置为关闭状态，关闭状态由 PSS1AC1:PSS1AC0 和 PSS1BD1:PSS1BD0 位（ECCP1AS3:ECCP1AS0）指定。每对引脚（P1A/P1C 和 P1B/P1D）可以设置为驱动高电平、驱动低电平或者三态 （不驱动）。ECCP1ASE 位 （ECCP1AS<7>）也被置 1，以保持增强型 PWM 输出为关闭状态。

当关闭事件发生时，ECCP1ASE 位由硬件置 1。如果自动重启功能没有使能，当关闭条件被清除时，ECCP1ASE 位由固件清零。如果自动重启被使能，当自动关闭起因被清除时， ECC1PASE 位自动清零。

如果ECCP1ASE位在PWM周期开始时被置1，则PWM 输出在整个 PWM 周期保持关闭状态。当 ECCP1ASE 位被清零时，PWM 输出将在下一个 PWM 周期开始时恢复正常操作。

注：	当关闭条件有效时，禁止写 ECCP1ASE 位。

寄存器 17-2： ECCPxDEL：ECCPx 死区延时寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **PxRSEN：**PWM 重启使能位

1 = 自动关闭时，一旦关闭事件消失， ECCPxASE 位自动清零； PWM 自动重启
0 = 自动关闭时， ECCPxASE 必须用软件清零以重启 PWM

bit 6-0 **PxDC6:PxDC0：**PWM 延时计数位

延时时间，指的是 PWM 信号转换为有效的预定时间和实际时间之间的时间差，单位为 FOSC/4 （4 * TOSC）。

寄存器 17-3： ECCPxAS：ECCPx 自动关闭配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **ECCPxASE**：ECCPx 自动关闭事件状态位
0 = ECCPx 输出正常工作
1 = 发生了关闭事件； ECCPx 输出为关闭状态

bit 6-4 **ECCPxAS2:ECCPxAS0**：ECCPx 自动关闭源选择位
000 = 禁止自动关闭
001 = 比较器 1 的输出
010 = 比较器 2 的输出
011 = 比较器 1 或比较器 2
100 = FLT0
101 = FLT0 或比较器 1
110 = FLT0 或比较器 2
111 = FLT0 或比较器 1 或比较器 2

bit 3-2 **PSSxAC1:PSSxAC0**：引脚 A 和 C 关闭状态控制位
00 = 驱动引脚 A 和 C 为 0
01 = 驱动引脚 A 和 C 为 1
1x = 引脚 A 和 C 为三态

bit 1-0 **PSSxBD1:PSSxBD0**：引脚 B 和 D 关闭状态控制位
00 = 驱动引脚 B 和 D 为 0
01 = 驱动引脚 B 和 D 为 1
1x = 引脚 B 和 D 为三态

17.4.7.1 自动关闭和自动重启

自动关闭功能可以配置为允许在关闭事件后自动重启模块。这可通过将 ECCP1DEL 寄存器的 P1RSEN 位（ECCP1DEL<7>）置 1 使能。

在关闭模式下，如果 P1RSEN = 1 （图 17-10），只要导致关闭的起因持续不变，ECCP1ASE 位将保持置 1。当关闭条件清除时，ECCP1ASE 位清零。如果 P1RSEN = 0（图 17-11），一旦关闭条件发生，ECCP1ASE 位将保持置 1 直到被软件清零。一旦 ECCP1ASE 位被清零，增强型 PWM 将在下一个 PWM 周期的开始重新工作。

注：	当关闭条件有效时，禁止写 ECCP1ASE 位。

与 P1RSEN 位的设置无关，如果自动关闭源为比较器之一，关闭条件是一个电平值。只要关闭的条件一直持续， ECCP1ASE 位就不能被清零。

可以通过写 1 到 ECCP1ASE 位，强制器件为自动关闭模式。

17.4.8 启动注意事项

当 ECCP 模块工作在 PWM 模式时，必须在 PWM 输出引脚上外接适当的上拉和 / 或下拉电阻。当单片机退出复位状态时，所有 I/O 引脚呈高阻状态。外部电路必须保持功率开关器件处于截止状态，直到单片机将 I/O 引脚驱动为适当的信号电平，或者激活 PWM 输出为止。

CCP1M1:CCP1M0 位（CCP1CON<1:0>）允许用户为每一对 PWM 输出引脚 （P1A/P1C 和 P1B/P1D）选择 PWM 输出信号为高电平有效或低电平有效。 PWM 输出极性必须在 PWM 引脚配置为输出之前选择。由于可能导致应用电路的损坏，因此不推荐在 PWM 引脚配置为输出的同时改变极性配置。

当 PWM 模块初始化时，P1A、P1B、P1C 和 P1D 输出锁存器可能不在正确的状态。这样在使能 ECCP 模块的同时使能 PWM 引脚为输出，可能损坏应用电路。应首先将 ECCP 模块配置为正确的输出模式并经过一个完整的 PWM 周期之后，再将 PWM 引脚设置为输出。当第二个 PWM 周期开始时， TMR2IF 位置 1 表明一个完整的 PWM 周期结束了。

图 17-10： **PWM 自动关闭 （P1RSEN = 1，使能自动重启）**

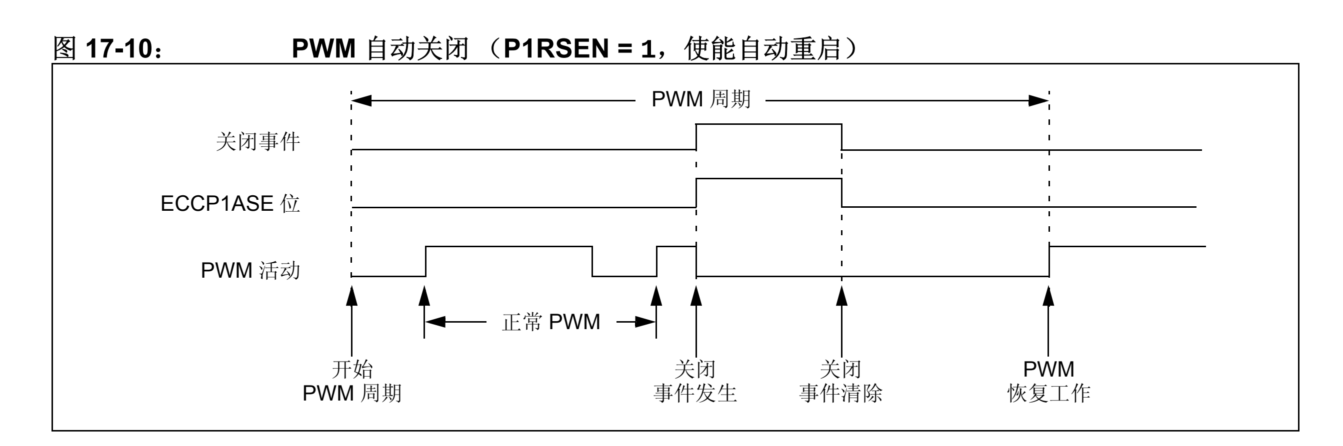
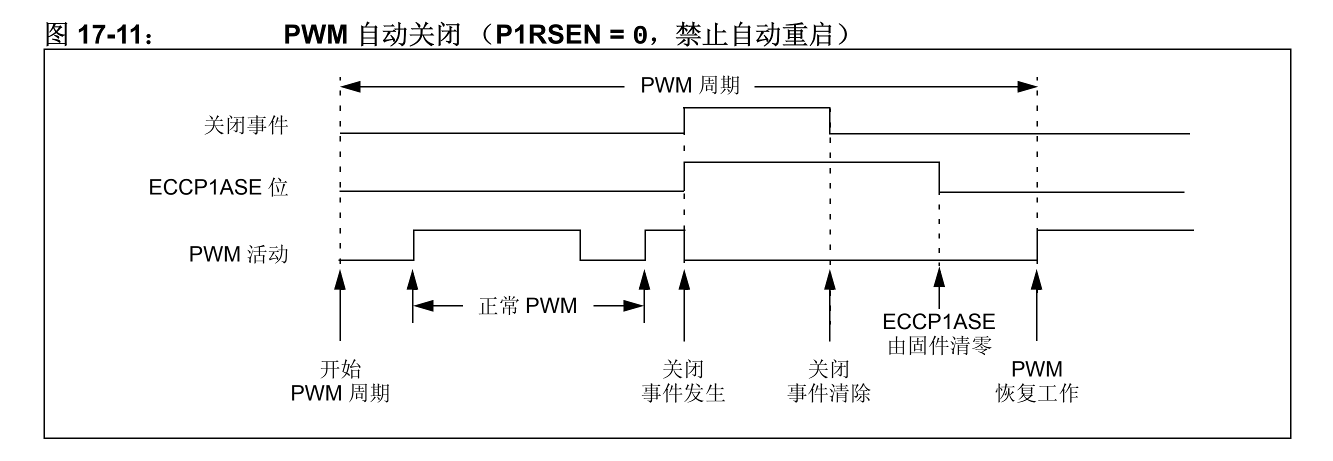


图 17-11： **PWM 自动关闭 （P1RSEN = 0，禁止自动重启）**



 超前信息

17.4.9 设置 PWM 工作模式

如果要将 ECCPx 模块配置成工作于 PWM 模式，可采用以下步骤：

1. 通过将相应的 TRIS 位置 1，配置 PWM 引脚 PxA 和 PxB（以及 PxC 和 PxD，如果使用）为输入。
2. 通过装载 PR2 （PR4）寄存器设置 PWM 周期。
3. 通过装载恰当的值到CCPxCON寄存器将ECCPx 模块设置为需要的 PWM 模式和配置：
   - 用 PxM1:PxM0 位选择输出配置和方向。
   - 用CCPxM3:CCPxM0位选择PWM输出信号的极性。
4. 通过装载 CCPRxL 寄存器和 CCPxCON<5:4> 位设置 PWM 占空比。
5. 对于自动关闭：
   - 禁止自动关闭； ECCP1ASE = 0
   - 配置自动关闭源
   - 等待运行条件
6. 对于半桥输出模式，通过装载恰当的值到 ECCPxDEL<6:0> 设置死区延时。
7. 如果需要自动关闭操作，装载ECCPxAS寄存器：
   - 使用 ECCPxAS2:ECCPxAS0 位选择自动关闭源。
   - 使用PSSxAC1:PSSxAC0和PSSxBD1:PSSxBD0 位选择 PWM输出引脚在关闭时的状态。
   - 将 ECCPxASE 位 （ECCPxAS<7>）置 1。
8. 如果需要自动重启，将PxRSEN位（ECCPxDEL<7>）置 1。
9. 配置和启动 TMRx （TMR2 或 TMR4）：
   - 通过清零 TMRxIF 位 （对于 Timer2 是 PIR1<1>，对于 Timer4 是 PIR3<3>）清零 TMRx 中断标志位。
   - 通过装载 TxCKPS 位 （TxCON<1:0>）设置 TMRx 预分频值。
   - 通过将 TMRxON 位 （TxCON<2>）置 1 使能 Timer2 （或 Timer4）。
10. 在新的 PWM 周期开始后使能 PWM 输出：
   - 等待直到 TMRx 溢出 （TMRxIF 位置 1）。
   - 通过清零相应的 TRIS 位，使能 ECCPx/PxA、 PxB、 PxC 和 / 或 PxD 引脚为输出。
   - 清零 ECCPxASE 位 （ECCPxAS<7>）。

17.4.10 复位的影响

上电复位及后续的复位都将强制所有端口为输入模式，并强制 CCP 寄存器为复位状态。

这将强制增强型 CCP 模块复位到与标准 CCP 模块兼容的状态。

表 17-5： 与 ECCP 模块和 TIMER1 到 TIMER4 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	61
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	61
TRISD[1]	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	61
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	61
TRISG	TRISA6[2]	TRISG6[2]	TRISG5[2]	TRISG4	TRISG3[2]	TRISG2[2]	TRISG1[2]	TRISG0[2]	61
TRISH[2]	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	61
TMR1L	Timer1 寄存器的低字节								60
TMR1H	Timer1 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	60
TMR2	Timer2 寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
PR2	Timer2 周期寄存器								60
TMR3L	Timer3 寄存器的低字节								60
TMR3H	Timer3 寄存器的高字节								60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	60
TMR4	Timer4 寄存器								62
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	62
PR4	Timer4 周期寄存器								62
CCPRxL[3]	捕捉 / 比较 /PWM 寄存器 x 的低字节								60
CCPRxH[3]	捕捉 / 比较 /PWM 寄存器 x 的高字节								60
CCPxCON[3]	PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0	60
ECCPxAS[3]	ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0	60, 63
ECCPxDEL[3]	PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0	63

图注： — = 未用 （读为 0）， r = 保留。 ECCP 操作中不使用阴影单元。

注 1： 仅适用于 64 引脚器件。

2： 寄存器和 / 或特定位在 64 引脚器件上未实现。

3： 所有增强型 CCP 模块中具有该名称的功能相同寄存器的通称，其中 “x” 代表特定模块 （ECCP1、 ECCP2 或 ECCP3）。通称相同的所有寄存器的位分配和复位值相同。

18.0 以太网模块

所有 PIC18F97J60 系列器件都具有嵌入式以太网控制器模块。这是一个完整的连接解决方案，完全实现了介质访问控制（Media Access Control，MAC）和物理层收发器（Physical Layer Transceiver，PHY）模块。只需使用两个脉冲变压器和一些无源元件就可以将单片机直接与以太网相连。

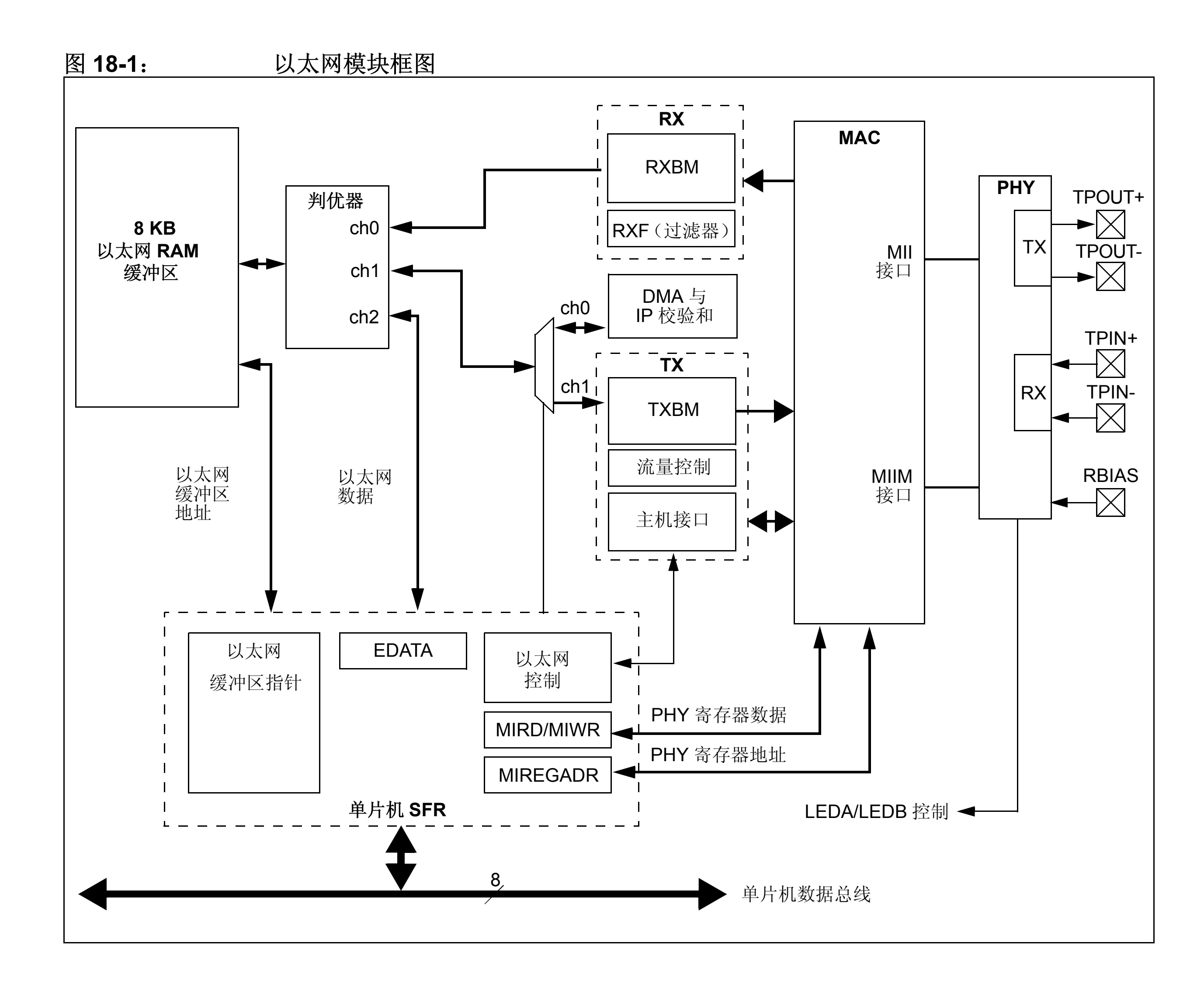
以太网模块符合 IEEE 802.3 中所有关于双绞线网络 10-BaseT 连接的规定。它在内部实现了一些数据包过滤机制，用以限制传入的数据包。它还提供了一个用于实现快速数据吞吐的内置 DMA 模块，以及硬件 IP 校验和计算。另外还提供了两个用于指示链路和网络活动的 LED 输出。

图 18-1 给出了该模块的简化框图。

以太网模块包含以下五个主要功能模块：

1. PHY 收发器模块——对传输双绞线接口上的模拟数据进行编码和解码，并通过网络发送和接收它。
2. MAC 模块——实现了符合 IEEE 802.3 规范的 MAC 逻辑，并提供了用以控制 PHY 的 MIIM（Media Independent Interface Management）。
3. 独立的 8 KB RAM 缓冲区——用于存储已经接收和将要发送的数据包。
4. 判优器——在单片机内核、DMA、发送和接收模块发出请求时，控制对 RAM 缓冲区的访问。
5. 寄存器接口——用作以太网模块和单片机特殊功能寄存器（SFR）之间命令和内部状态信号的译码器。

图 18-1： 以太网模块框图



18.1 物理接口和外部连接

18.1.1 信号和电源接口

PIC18F97J60 系列器件均提供一个 4 引脚的以太网模块专用信号接口。这些引脚没有与其他的单片机或外设功能复用，因此不必考虑潜在的器件配置冲突问题。这些引脚是：

- TPIN+：双绞线差分信号正向输入
- TPIN-：双绞线差分信号负向输入
- TPOUT+：双绞线差分信号正向输出
- TPOUT-：双绞线差分信号负向输出

目前尚无法与外部以太网控制器或 MAC/PHY 子系统之间进行数字以太网数据的收发。

除了信号连接之外，以太网模块还具有自己独立的电压源和地供内部 PHY 模块使用。对于接收器、发送器和发送器的内部 PLL，都提供独立的电压源和地引脚；接收器是 V_{DDRX} 和 V_{SSRX}，发送器是 V_{DDTX} 和 V_{SSTX}，发送器的内部 PLL 是 V_{DDPLL} 和 V_{SSPLL}。虽然它们的电压要求与单片机的 V_{DD} 和 V_{SS} 相同，但在内部并未将这些引脚相连。为使以太网模块正常工作，还必须将供电电压和地与这些引脚相连接。此外，还需要在外部将单片机的电源和地引脚连接到同一电源或接地点。

除了独立的电源连接，PHY 模块还具有独立的偏置电流输入引脚 RBIAS。为使收发器正常工作，必须向 RBIAS 引脚施加偏置电流。

18.1.2 LED 配置

PHY 模块提供了独立的输出来驱动标准以太网指示器 LEDA 和 LEDB。LED 输出与 PORTA 引脚 RA0 和 RA1 复用。通过置 1 配置位 ETHLED（寄存器 24-6，CONFIG3H<2>），可以将它们配置为 LED 输出。当配置为 LED 输出时， RA0/LEDA 和 RA1/LEDB 具有足够的驱动能力 （8mA），可以直接驱动 LED。必须始终将这两个引脚配置为向 LED 提供电流（源）。用户还必须通过清零 TRISA<1:0> 将这两个引脚配置为输出。

这两个 LED 可以分别配置为自动显示链路状态、RA/TX 活动等。它们还具有可配置延时功能，可以对短时间事件 （如单个数据包发送）的 LED 闪烁时间进行延长，以方便人们查看。相关选项由 PHLCON 寄存器 （寄存器 18-14）控制。表 18-1 列出了闪烁延时的典型值。

表 18-1： LED 闪烁延时长度

延时长度	典型延时 （ms）
T_{NSTRCH} （正常）	40
T_{MSTRCH} （中等）	70
T_{LSTRCH} （长）	140

18.1.3 振荡器要求

以太网模块的设计工作频率为 25 MHz。该频率由单片机主时钟 （在 OSC1 和 OSC2 引脚连接晶振，或者在 OSC1 引脚连接外部时钟源）提供。不能为模块配置一个不同的时钟源。

18.1.3.1 起振定时器

以太网模块具有一个起振定时器，它独立于单片机的起振定时器 OST，用以保证模块在 PHY 模块的 PLL 稳定以后再开始工作。清零模块使能位 ETHEN （ECON2<5>）将清零 PHYRDY 状态位 （ESTAT<0>）。将 ETHEN 位置 1 会使起振定时器开始计数。当定时器定时结束时，经过 1 ms 之后， PHYRDY 位会自动置 1。

应用软件应一直查询 PHYRDY 位，以确定何时可以启动正常的以太网操作。

 超前信息

18.1.4 磁性元件、端接元件和其他外部元件

为实现以太网接口，以太网模块需要外部连接一些标准元件。应按图 18-2 所示来连接这些元件。

PHY 模块中的内部模拟电路要求在 RBIAS 和地之间连接一个外部电阻 REBIAS。该电阻会影响 TPOUT+/- 信号的幅度，应布置在尽可能靠近芯片的位置，并避免与信号走线相邻，以防止噪声耦合到引脚中，从而对发送操作造成不利影响。建议该电阻采用表贴元件。

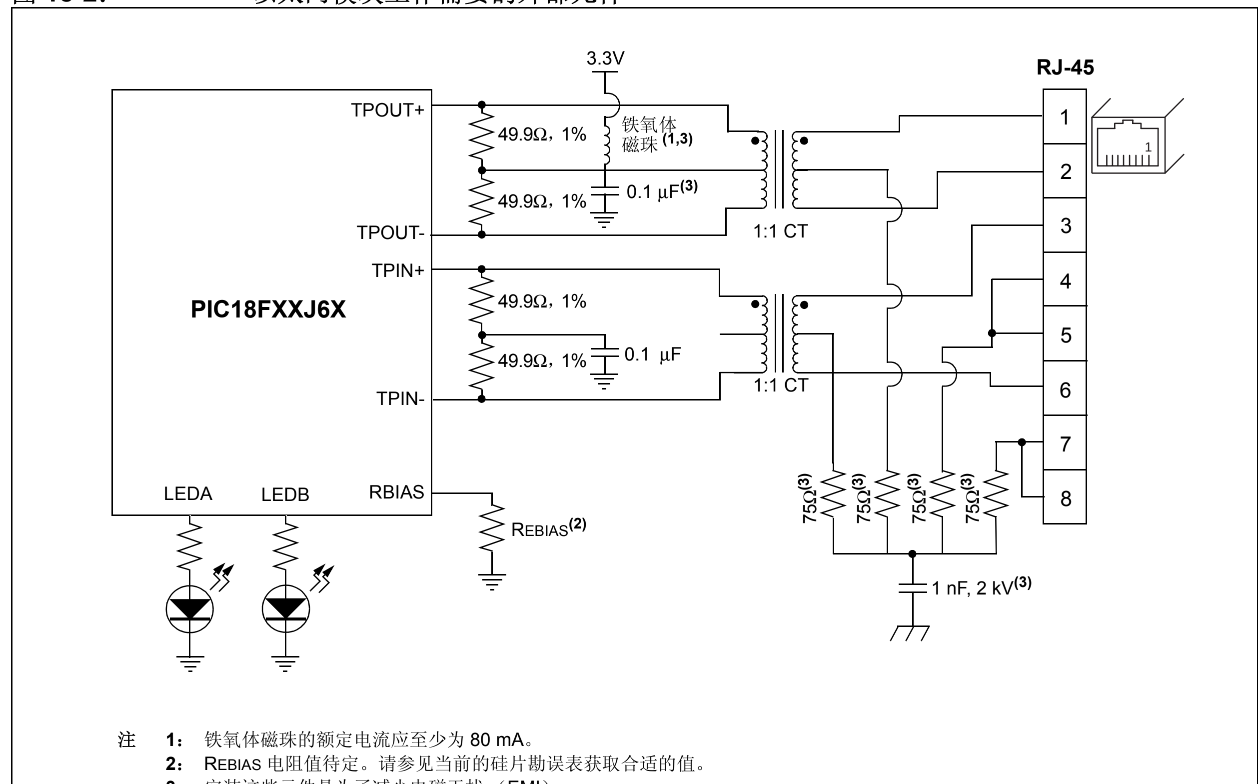
在 TPIN+/TPIN- 和 TPOUT+/TPOUT- 引脚，需要连接用于以太网操作的 1:1 中心抽头脉冲变压器。当使能以太网模块时，两个 TPOUT 引脚会有持续的电流流入。当 PHY 在发送数据时，通过改变 TPOUT+ 和 TPOUT- 的相对电流大小，将在以太网电缆上产生一个差分电压。

建议不要在 TPOUT 引脚和以太网变压器 （未显示）之间，安放针对 TPOUT 接口的共模扼流器。如果要使用共模扼流器来减少 EMI 辐射，应将它安放在以太网变压器和 RJ-45 连接器的 1、2 引脚之间。许多以太网变压器模块都在同一器件封装内包含了共模扼流器。变压器应该至少达到表 27-28 中规定的绝缘等级，才能防止静电的干扰，并符合 IEEE 802.3 中的绝缘要求（关于变压器的具体要求，请参见**第 27.5 节 “以太网规范和要求”**）。此外，发送和接收接口均需要另外使用两个电阻和一个电容来正确地端接传输线，以最大限度地减少信号反射。

所有的电源引脚必须外接于同一电源。类似地，所有参考地也必须外接于同一接地点。对于每个 VDD 和 VSS 引脚对，都应该在尽可能靠近引脚的位置安放一个 0.1 μF 的陶瓷旁路电容。

由于双绞线接口工作时需要相对较高的电流，因此所有接线都应该尽可能的短。电源线的线径要合适，以减少阻性损耗。如果无法使用较短的差分数据线，则应将其配接成具有 50Ω 的特性阻抗。

图 18-2： 以太网模块工作需要的外部元件



注 1： 铁氧体磁珠的额定电流应至少为 80 mA。
2： REBIAS 电阻值待定。请参见当前的硅片勘误表获取合适的值。
3： 安装这些元件是为了减少电磁干扰 （EMI）。

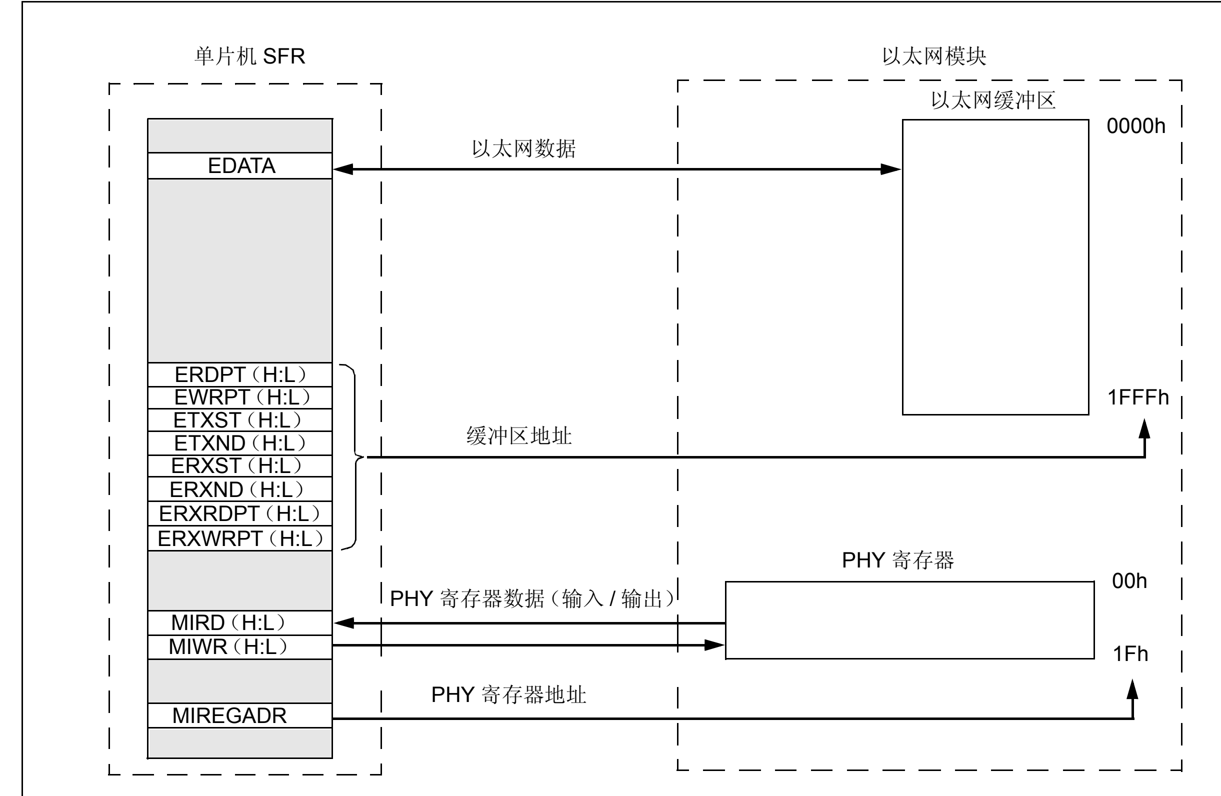
18.2 以太网缓冲区和寄存器空间

以太网模块使用三个独立的存储器空间：

- 以太网 RAM 缓冲区——用于存储接收到的数据和准备进行发送的数据。
- 一组 8 位的特殊功能寄存器（SFR）——用于控制模块，以及在模块和单片机内核之间来回传递数据。
- 一组独立的 16 位 PHY 寄存器——专门用于 PHY 控制和状态报告。

以太网缓冲区和 PHY 控制寄存器全部位于以太网模块内，单片机不能直接对其进行访问。在以太网和单片机之间，数据通过使用缓冲区和映射在单片机 SFR 空间中的指针寄存器来传送。图 18-3 给出了 SFR 与以太网模块存储器空间之间的关系。

图 18-3： 单片机和以太网存储器空间之间的关系



注 1： 单片机的 SFR 不是按照它们在数据存储器空间中的存放顺序显示的。存储器区域未按比例显示。

 超前信息

18.2.1 以太网缓冲区和缓冲区指针寄存器

以太网缓冲区中包含了供以太网控制器使用的发送存储器和接收存储器。整个缓冲区为 8 KB 大小，分为单独的接收缓冲区空间和发送缓冲区空间。不论以太网模块是否使能，总是可以通过 EDATA 和以太网指针 SFR 来对缓冲区进行访问。发送和接收存储器的大小和位置可以使用以太网 SFR 空间中的指针自由定义。

18.2.1.1 接收缓冲区

接收缓冲区实际上是一个由硬件管理的循环 FIFO 缓冲区。寄存器对 ERXSTH:ERXSTL 和 ERXNDH:ERXNDL 用作指针，用来定义缓冲区在存储器中的大小和位置。ERXST和ERXND指向的字节都包含在FIFO缓冲区中。

在从以太网接口接收到数据字节时，会将它们顺序地写入接收缓冲区。不过，在写入 ERXND 所指向的存储单元之后，硬件将自动将接收数据的下一字节写入 ERXST 所指向的存储单元。从而，接收硬件永远不会将数据写入 FIFO 边界之外的存储单元。

当接收逻辑禁止时，用户可以设定 ERXST 和 ERXND 指针。当接收逻辑使能，即 ERXEN （ECON1<2>）置 1 时，绝对不能修改指针。

ERXWRPTH:ERXWRPTL 寄存器定义在 FIFO 中，硬件将接收的数据字节写入的位置。该指针为只读指针，每当收到新的数据包 （如果未被过滤掉），硬件都会自动更新它。指针可用来判断 FIFO 中有多少剩余空间。

ERXRDPT 寄存器定义 FIFO 中禁止接收硬件写入的单元。在正常操作下，接收硬件会一直写入数据，直到 ERXRDPT 所指向的存储单元 （不包括该单元）。如果 FIFO 中已充满数据，而新的数据继续到达，硬件并不会用新数据覆盖前面接收到的数据。相反，它将丢弃新收到的数据，而保留旧的数据。为了能连续接收新数据，应用程序必须在每次处理完部分或全部旧的接收数据之后周期性地递增指针。

18.2.1.2 发送缓冲区

8 KB 存储器中未被定义为接收 FIFO 缓冲区的部分被视为发送缓冲区。应用程序需要负责管理数据包在发送缓冲区中的存储位置。每当应用程序要发送数据包时，它需要设定 ETXST 与 ETXND 指针，使之指向发送缓冲区中的待发送数据包。硬件并不检查起始和结束地址是否与接收缓冲区重叠。为防止缓冲区数据遭到破坏，在 ETXST 和 ETXND 指针与接收缓冲区重叠时，或 ETXND 指针与接收缓冲区太靠近时 ，固件都不应发送数据包。更多信息，请参见**第 18.5.2 节 “发送数据包”**。

18.2.1.3 读写缓冲区

以太网缓冲区的内容可通过 EDATA 寄存器进行访问，EDATA 寄存器是单片机数据总线到缓冲区的一个窗口。该窗口的位置由 ERDPT 或者 EWRPT 指针决定，具体使用哪个指针取决于所执行的操作。例如，写 EDATA 将导致在 EWRPTH:EWRPTL 当前指示的地址处对以太网缓冲区进行写操作。类似地，将 EDATA 中的内容移动到另外一个寄存器时，实际上移动的是 ERDPTH:ERDPTL 所指地址处的缓冲区内容。

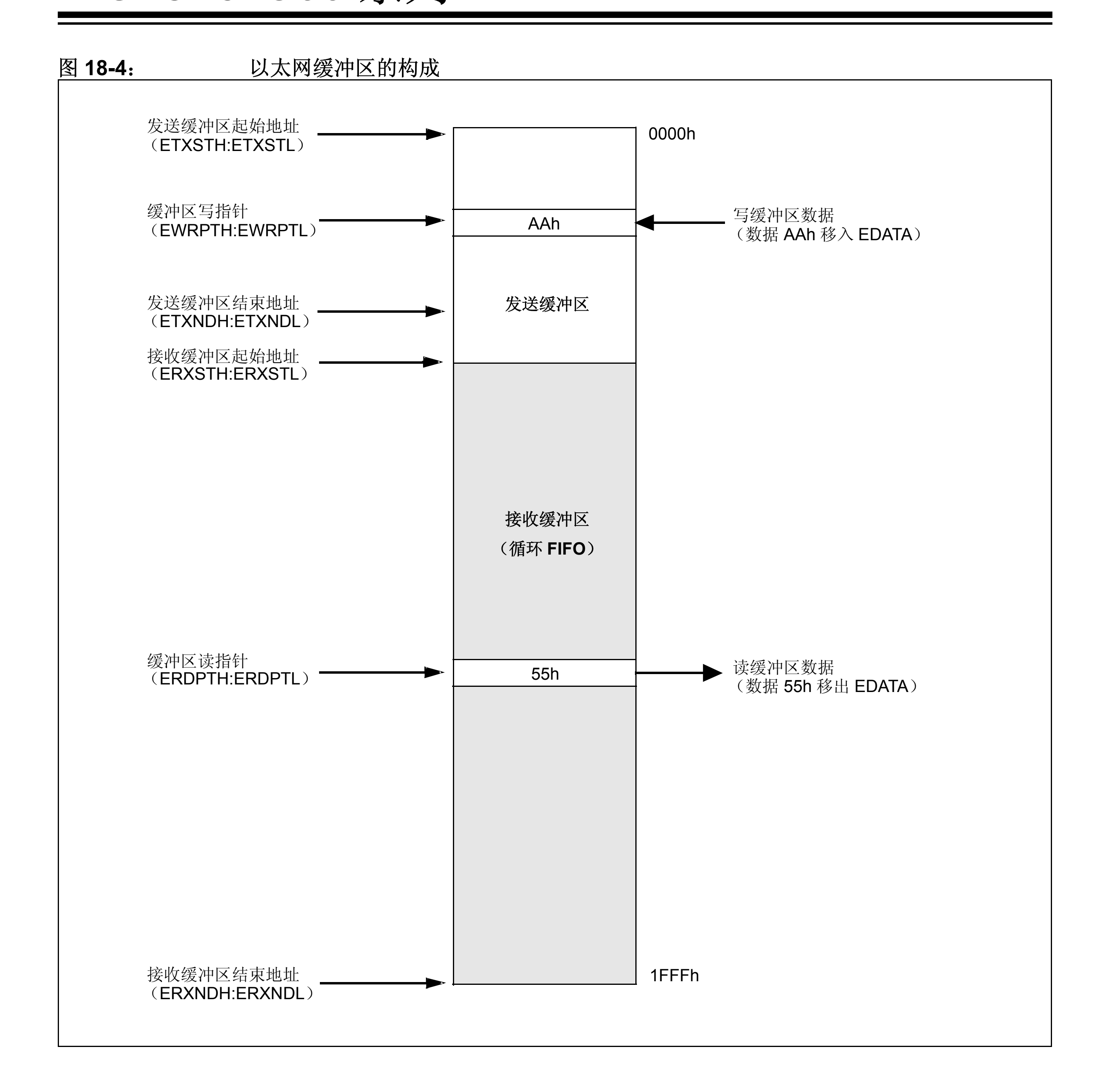
将 AUTOINC 位（ECON2<7>）置 1 会使相关的读或写指针在每次读或写操作之后递增一位地址。这使得不必在每次读或写操作之后不断地更新指针，从而简化了连续读写操作的步骤。

当顺序地读接收缓冲区时，在接收缓冲区的底部会出现折回现象。根据 ERXNDH:ERXNDL 寄存器中的地址对 EDATA 进行读操作，将导致 ERDPTH:ERDPTL 递增从而与 ERXSTH:ERXSTL 寄存器中所包含的值相等。当顺序地写缓冲区时，将不会出现折回现象。

18.2.1.4 缓冲区 DMA 访问

当计算校验和时，内部集成的 DMA 控制器必须读缓冲区；当复制存储器内容时，它必须读写缓冲区。DMA 同样遵循前面所描述的折回规则。当顺序地读时，在接收缓冲区的底部将发生折回。而进行写操作时则不会出现折回现象。更多信息，请参见**第 18.9 节 “直接存储器访问控制器”**。

图 18-4：　以太网缓冲区的构成



18.2.2 SFR 和以太网模块

和其他外设一样，以太网模块的直接控制是通过一组 SFR 来完成的。因为数量众多，其中的大多数寄存器位于单片机数据存储器空间中 Bank 14 的下半部分。

以太网模块的 5 个关键 SFR 位于单片机 Bank 15 的常用 SFR 区，以便可以进行快速访问。它们是：

- ECON1
- EDATA
- EIR
- 以太网缓冲区读指针对 （ERDPTH 和 ERDPTL）

ECON1 将留在下一节中与其他以太网控制寄存器一起进行讨论。EDATA 和 ERDPTH:ERDPTL 分别是以太网数据缓冲区寄存器及其读操作指针（见**第 18.2.1 节“以太网缓冲区和缓冲区指针寄存器”**）。 EIR 是以太网中断结构的一部分，将在**第 18.3 节 “以太网中断”**中对它进行说明。

Bank 14 中的大部分以太网 SFR 都用作指针寄存器，用来指定专用以太网缓冲区中的地址，供数据存储和检索之用。其他寄存器则存储用于数据包格式屏蔽或校验和操作的信息。有几个寄存器用于控制整体的模块操作和具体的 MAC 和 PHY 功能。

18.2.3 以太网控制寄存器

ECON1 寄存器 （寄存器 18-1）用来控制模块的主要功能。接收使能位、发送请求位和 DMA 控制位都位于其中。 ECON2 寄存器 （寄存器 18-2）用来控制模块的其他上层功能。ESTAT寄存器（寄存器 18-3）用来报告模块和以太网通信的高级状态。

不论模块是否使能，总是可以对名称含有 E 前缀的以太网 SFR 进行访问。

寄存器 18-1： ECON1：以太网控制寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **TXRST**：发送逻辑复位位
1 = 发送逻辑保持在复位状态
0 = 正常操作

bit 6 **RXRST**：接收逻辑复位位
1 = 接收逻辑保持在复位状态
0 = 正常操作

bit 5 **DMAST**：DMA 启动和忙状态位
1 = DMA 复制或校验和操作正在进行中 （由软件置 1、由硬件或软件清零）
0 = DMA 硬件空闲

bit 4 **CSUMEN**：DMA 校验和使能位
1 = DMA 硬件计算校验和
0 = DMA 硬件复制缓冲存储区内容

bit 3 **TXRTS**：发送请求位
1 = 发送逻辑正在尝试发送数据包 （由软件置 1、由硬件或软件清零）
0 = 发送逻辑空闲

bit 2 **RXEN**：接收使能位
1 = 通过当前过滤器配置的数据包将被写入接收缓冲区
0 = 所有接收到的数据包将被硬件丢弃

bit 1-0 未用：读为 0

寄存器 18-2： ECON2：以太网控制寄存器 2

R/W-1	R/W-0[1]	R/W-0	U-0	U-0	U-0	U-0	U-0
AUTOINC	PKTDEC	ETHEN	—	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **AUTOINC：**缓冲区指针自动递增使能位
1 = 在读或写 EDATA 时自动递增 ERDPT 或 EWRPT
0 = 访问 EDATA 后不自动改变 ERDPT 和 EWRPT

bit 6 **PKTDEC：**数据包递减位
1 = 将 EPKTCNT 寄存器减 1
0 = 不改变 EPKTCNT

bit 5 **ETHEN：**以太网模块使能位
1 = 使能以太网模块
0 = 禁止以太网模块

bit 4-0 未用：读为 0

注 1：一旦被置 1，该位会自动清零。

寄存器 18-3： ESTAT：以太网状态寄存器

U-0	R/C-0	U-0	R/C-0	U-0	R-0	R/C-0	R-0
—	BUFER	—	LATECOL	—	RXBUSY	TXABRT	PHYRDY
bit 7							bit 0

图注：			
R = 可读位	C = 可清零位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 未用：读为 0

bit 6 **BUFER：**以太网缓冲区错误状态位
1 = 以太网读或写操作发生了缓冲区错误 （数据溢出或数据不足）
0 = 未发生缓冲区错误

bit 5 未用：读为 0

bit 4 **LATECOL：**迟冲突错误位
1 = 在发送完 64 字节后发生冲突
0 = 发送 64 字节后无冲突发生

bit 3 未用：读为 0

bit 2 **RXBUSY：**接收忙位
1 = 接收逻辑正在接收数据包
0 = 接收逻辑空闲

bit 1 **TXABRT：**发送中止错误位
1 = 发送请求被中止
0 = 无发送中止错误

bit 0 **PHYRDY：**以太网 PHY 时钟就绪位
1 = 以太网 PHY 起振定时器 （OST）定时结束， PHY 就绪
0 = 以太网 PHY 起振定时器仍在计数， PHY 未就绪

18.2.4 MAC 和 MII 寄存器

这些 SFR 用于控制 MAC 的操作，并通过 MIIM 控制 PHY 的操作。MAC 和 MII 寄存器占用数据地址 E80h-E85h、 E8Ah，以及从 EA0h 至 EB9h。

虽然 MAC 和 MII 寄存器出现在单片机的常规存储器映射中，但它们实际上是内嵌在 MAC 模块中的。主机接口逻辑会转换单片机数据 / 地址总线数据，以便能够访问这些寄存器。主机接口逻辑对固件访问 MAC 和 MII 特殊功能寄存器的方式作出了一些限制。请参见下方的“注”。

注	1： 只有在以太网模块使能时，即 ETHEN （ECON2<5>）= 1 时，才可以对 MAC 和 MII 特殊功能寄存器 （SFR）进行访问。 2： 不支持背对背访问 MAC 或 MII 寄存器。在寻址 MAC 或 MII 寄存器的任意两条指令之间，必须执行至少一条 NOP 或其他指令。

三个 MACON 寄存器控制具体的 MAC 操作和数据包配置操作。它们如寄存器 18-4 至寄存器 18-6 所示。

MII 寄存器用于控制 MIIM 接口，并用作与 PHY 寄存器之间的通信通道。请参见寄存器 18-7、寄存器 18-8 和寄存器 18-9。

寄存器 18-4： MACON1：MAC 控制寄存器 1

U-0	U-0	U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-5 未用：读为 0

bit 4 保留：保持为 0

bit 3 **TXPAUS**：暂停控制帧发送使能位
1 = 允许 MAC 发送暂停控制帧 （全双工模式中用于流量控制）
0 = 禁止暂停帧发送

bit 2 **RXPAUS**：暂停控制帧接收使能位
1 = 当接收到暂停控制帧时禁止发送 （正常操作）
0 = 忽略接收到的暂停控制帧

bit 1 **PASSALL**：所有接收帧通过使能位
1 = MAC 接收到的控制帧如果未被过滤掉则写入接收缓冲区
0 = 控制帧在 MAC 处理后被丢弃 （正常操作）

bit 0 **MARXEN**：MAC 接收使能位
1 = 允许 MAC 接收数据包
0 = 禁止接收数据包

寄存器 18-5： MACON3：MAC 控制寄存器 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-5 **PADCFG2:PADCFG0：**自动填充和 CRC 配置位

111 = 对所有短帧用 0 填充至 64 字节，然后追加一个有效的 CRC
110 = 对短帧不进行自动填充
101 = MAC将自动检测具有8100h类型字段的VLAN协议帧，并自动填充至64字节。如果该帧不是VLAN帧，则填充至 60 字节。填充之后，追加一个有效的 CRC。
100 = 对短帧不进行自动填充
011 = 对所有短帧用 0 填充至 64 字节，然后追加一个有效的 CRC
010 = 对短帧不进行自动填充
001 = 对所有短帧用 0 填充至 60 字节，然后追加一个有效的 CRC
000 = 对短帧不进行自动填充

bit 4 **TXCRCEN：**发送 CRC 使能位

1 = 不论PADCFG状态如何，MAC将对所有发送的帧追加一个有效的CRC。如果PADCFG指定追加一个有效的 CRC，则 TXCRCEN 必须置 1。
0 = MAC 不会追加 CRC。对最后 4 个字节进行校验，如果不是有效的 CRC，将在发送状态向量中给予报告。

bit 3 **PHDREN：**专用报头使能位

1 = 发送给 MAC 的帧包含一个 4 字节的专用报头，在计算 CRC 时不包括该报头。
0 = 不存在专用报头。 CRC 将涵盖所有数据 （正常操作）。

bit 2 **HFRMEN：**超大帧使能位

1 = 允许发送和接收任意大小的帧
0 = 帧如果大于 MAMXFL，则中止发送或接收。

bit 1 **FRMLNEN：**帧长度检查使能位

1 = 对发送和接收帧的类型 / 长度字段进行检查。如果是长度字段，则比较帧的大小，并在发送 / 接收状态向量中报告不匹配状态。
0 = 不将帧长度和类型 / 长度字段进行比较。

bit 0 **FULDPX：**MAC 全双工使能位

1 = MAC 将工作于全双工模式。 PDPXMD 位也必须置 1。
0 = MAC 将工作于半双工模式。 PDPXMD 位也必须清零。

寄存器 18-6： MACON4：MAC 控制寄存器 4

U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R-0	R-0
—	DEFER	BPEN	NOBKOFF	—	—	r	r
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7　未用：读为 0

bit 6　**DEFER**：延迟发送使能位 （仅适用于半双工模式）
1 = 当介质被占用时， MAC 将无限期等待，直至介质空闲才尝试发送 （该设置符合 802.3 规范）
0 = 当介质被占用时， MAC 在等待时间达到极限值后将中止发送

bit 5　**BPEN**：背压期间无后退 （Backoff）使能位 （仅适用于半双工模式）
1 = 在背压期间意外发生冲突之后， MAC 将立即开始重发
0 = 在背压期间意外发生冲突之后， MAC 将延时一段时间后 （延时时间采用二进制指数后退算法计算）再尝试重发 （正常操作）

bit 4　**NOBKOFF**：无后退使能位 （仅适用于半双工模式）
1 = 在发生任何冲突后， MAC 将立即开始重发
0 = 在发生任何冲突后，MAC 将延时一段时间后（延时时间采用二进制指数后退算法计算）再尝试重发 （正常操作）

bit 3-2　未用：读为 0

bit 1-0　保留：保持为 0

寄存器 18-7： MICON：MII 控制寄存器

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
RSTMII	—	—	—	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7　**RSTMII**：MII 管理模块复位位
1 = MII 管理模块保持在复位状态
0 = 正常操作

bit 6-0　未用：读为 0

寄存器 18-8： MICMD：MII 命令寄存器

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	MIISCAN	MIIRD
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-2 未用：读为 0

bit 1 **MIISCAN**：MII 扫描使能位
1 = 连续读 MIREGADR 中的 PHY 寄存器，并将数据写入 MIRD
0 = 不在进行 MII 管理扫描操作

bit 0 **MIIRD**：MII 读使能位
1 = 读 MIREGADR 中的 PHY 寄存器一次，并将数据写入 MIRD
0 = 不在进行 MII 管理读操作

寄存器 18-9： MISTAT：MII 状态寄存器

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	r	NVALID	SCAN	BUSY
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-4 未用：读为 0

bit 3 保留：保持为 0
1 = 不存在链路伙伴，或者以太网电缆未连接
0 = 链路未失效

bit 2 **NVALID**：MII 管理读数据无效位
1 = MIRD 的内容尚未变为有效
0 = MII 管理读周期已经完成，并且 MIRD 已经更新

bit 1 **SCAN**：MII 管理扫描操作位
1 = 正在进行 MII 管理扫描操作
0 = 未进行 MII 管理扫描操作

bit 0 **BUSY**：MII 管理忙位
1 = 正在读或写 PHY 寄存器
0 = MII 管理接口空闲

18.2.5 PHY 寄存器

PHY 寄存器可对 PHY 模块进行配置和控制，并提供操作状态信息。所有 PHY 寄存器都是 16 位宽。

总共有 32 个 PHY 地址，但只实现了其中 9 个。对未实现单元进行写操作将被忽略，而对这些单元的所有读操作都将返回 0。所有保留的单元都应写为 0 ；当读取时应忽略其内容。

表 18-2 列出了所有已实现的 PHY 寄存器。关于主要 PHY 控制寄存器的描述，请参见寄存器 18-10 至寄存器 18-14。其他的 PHY 控制和状态寄存器在本章后续部分中讨论。

只有通过 MII 管理接口才可以对 PHY 寄存器进行访问。只有在 PHY 起振定时器定时结束，并且 PHYRDY 位（ESTAT<0>）置 1 之后，才可以对 PHY 寄存器进行读写。

18.2.5.1 PHSTAT 寄存器

PHSTAT1 与 PHSTAT2 寄存器中含有一些只读位，它们显示PHY模块操作的当前状态，特别是与网络其他部分的通信链路的状态。

PHSTAT1 寄存器（寄存器 18-11）中含有 LLSTAT 位；如果从上次读寄存器之后物理层链路断开，该位将清零并锁存为低电平。主机控制器可以周期性地查询该位，以确定链路失效的确切时间。如果未使用链路变化中断，那么该方法尤其有用。

PHSTAT1 寄存器还包含一个超时传输 （jabber）状态位。如果以太网控制器不间断地发送数据，使其他节点无法共享介质，那么就称之为超时传输。通常，超时传输状态表明本地控制器很可能违反了 IEEE 规范中的最大数据包长度的规定。该位锁存为高电平表示上次读寄存器之后发生了超时传输状态。

PHSTAT2 寄存器（寄存器 18-13）中含有指示 PHY 模块是否连入网络、是否在进行发送或接收的状态位。

18.2.5.2 PHID1 和 PHID2 寄存器

PHID1 和 PHID2 寄存器为只读寄存器。它们保存用以帮助鉴别以太网控制器的常数，对于进行调试可能会有帮助。其中包括：

- PHY 模块的部件号 （PPN5:PPN0）
- PHY 模块的版本号 （PREV3:PREV0）
- PHY 标识符，是 Microchip 公司组织唯一标识符（Organizationally Unique Identifier， OUI）的一部分 （OUI3:OUI24）

PHY 部件号和版本号是 PHID2 的一部分。 PHY 标识符的两个高字节位于 PHID1，其余部分位于 PHID2。表 18-2 给出了它们在寄存器中的确切位置。

版本信息也存储在 EREVID 中。这是一个只读控制寄存器，其中包含一个 5 位的标识符，指示器件的具体硅片版本号。

18.2.5.3 访问 PHY 寄存器

如上文所述，PHY 寄存器位于不同的存储器空间，且单片机不能直接访问它们。它们是通过以太网 SFR 存储区中一组专门的 MII 寄存器来进行寻址，这些寄存器实现了介质无关接口管理（Media Independent Interface Management， MIIM）。

其访问方式与以太网缓冲区的访问方式类似，但是使用独立的读和写缓冲区（MIRDH:MIRDL和MIWRH:MIWRL）及一个 5 位的地址寄存器（MIREGADR）。此外，还用到 MICMD 和 MISTAT 寄存器来控制读写操作。

读 PHY 寄存器：

1. 将要读的PHY寄存器的地址写入MIREGADR寄存器。
2. 将 MIIRD 位 （MICMD<0>）置 1。读操作开始，BUSY 位 （MISTAT<0>）置 1。
3. 等待 10.24 μs，然后查询 BUSY 位，确定操作是否完成。当 MAC 已获得寄存器的内容时，BUSY 位将自动清零。当 BUSY 位置 1 时，用户应用程序不能启动任何 MIISCAN 操作，或写 MIWRH 寄存器。
4. 清零 MIIRD 位。
5. 从 MIRDL 和 MIRDH 寄存器中读取 PHY 寄存器的全部 16 位。

写 PHY 寄存器：

1. 将要写的PHY 寄存器的地址写入MIREGADR寄存器。
2. 将要写数据的低 8 位写入 MIWRL 寄存器。
3. 将要写数据的高 8 位写入 MIWRH 寄存器。写 MIWRH 寄存器会自动启动 MII 事务，因此必须在写 MIWRL 之后才能写 MIWRH。 BUSY 位自动置 1。

在 MII 操作完成之后会开始写 PHY 寄存器，这需要 10.24 μs 的时间。当写操作完成后， BUSY 位会自动清零。在忙时，应用程序不能进行任何 MII 扫描或读操作。

当写 PHY 寄存器时，整个 16 位是一次性写入的，不能够选择性地对某些位进行写操作。如果只需要对寄存器中的某些位进行重新设定，控制器必须首先读取PHY寄存器，修改结果数据，然后将数据再写回 PHY 寄存器。

MAC 也可以配置为对 PHY 寄存器进行自动背对背读操作。要执行此扫描操作：

1. 将要扫描的 PHY 寄存器的地址写入 MIREGADR 寄存器。
2. 将 MIISCAN 位 （MICMD<1>）置 1。扫描操作开始， BUSY 位置 1。

MIISCAN 置 1 后，NVALID（MISTAT<2>）、SCAN 和 BUSY 位也都置 1。第一个读操作完成需要 10.24 μs。然后，后续的读操作会以相同的时间间隔执行，同时 MIRDL 和 MIRDH 寄存器会不断地自动更新，直至操作被取消为止。可以通过查询 NVALID 位判断第一个读操作何时完成。

没有可以用来确定 MIRD 寄存器何时更新的相关状态信息。因为一次只能读取一个 MII 寄存器，所以绝对不能认为 MIRDL 和 MIRDH 的值是同时从 PHY 中读出的。在扫描模式下，MIRDH 和 MIRDL 的值只有在 NVALID 清零之后才是有效的。 NVALID 会在第一个读序列完成后立即自动清零。

只要需要进行扫描操作， MIISCAN 都应保持置 1。在 MIISCAN 清零且最后一个读序列完成之后， BUSY 和 SCAN 位会自动清零。在 MIISCAN 置 1 时，不能对 MIREGADR 进行更新。

当正在进行扫描时，不应启动新的 PHY 操作，例如进行读操作或写 MIWRH 寄存器。可以通过清零 MIISCAN 位然后查询 BUSY 位来取消操作。在 BUSY 位清零之后，可以启动新的操作。

表 18-2： PIC18F97J60 系列 PHY 寄存器汇总

地址	名称	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
00h	PHCON1	PRST	PLOOPBK	—	—	PPWRSV	r	—	PDPXMD	r	—	—	—	—	—	—	—	`00-- 00-0 0--- ----`
01h	PHSTAT1	—	—	—	PHFDPX	PHHDPX	—	—	—	—	—	—	—	—	LLSTAT	JBSTAT	—	`---1 1--- ---- -00-`
02h	PHID1	PHY 标识符（OUI3:OUI18）= 0083h																`0000 0000 1000 0011`
03h	PHID2	PHY 标识符（OUI19:OUI24）= 000101						PHY P/N（PPN5:PPN0）= 00h						PHY 版本号（PREV3:PREV0）= 00h				`0001 0100 0000 0000`
10h	PHCON2	—	FRCLNK	TXDIS	r	r	JABBER	r	HDLDIS	r	r	r	r	r	r	r	r	`-000 0000 0000 0000`
11h	PHSTAT2	—	—	TXSTAT	RXSTAT	COLSTAT	LSTAT	DPXSTAT[1]	—	—	—	PLRITY	—	—	—	—	—	`--00 00x- --0- ----`
12h	PHIE	r	r	r	r	r	r	r	r	r	r	r	PLNKIE	r	r	PGEIE	r	`xxxx xxxx xx00 xx00`
13h	PHIR	r	r	r	r	r	r	r	r	r	r	r	PLNKIF	r	PGIF	r	r	`xxxx xxxx xx00 00x0`
14h	PHLCON	r	r	r	r	LACFG3:LACFG0				LBCFG3:LBCFG0				LFRQ1:LFRQ0		STRCH	r	`0011 0100 0010 001x`

图注： x = 未知， u = 不变， - = 未用（读为 0）， r = 保留（不能修改）。未使用阴影单元，读为 0。

注 1： 双工模式 / 状态位的复位值取决于 LED 到 LEDB 引脚的连接（更多详细信息，请参见**第 18.1.2 节 “LED 配置”**）。

超前信息

寄存器 18-10： PHCON1：PHY 控制寄存器 1

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0	R/W-0
PRST	PLOOPBK	—	—	PPWRSV	r	—	PDPXMD
bit 15							bit 8

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
r	—	—	—	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 15 **PRST**：PHY 软件复位位
1 = PHY 正在进行软件复位 （完成后自动复位为 0）
0 = 正常操作

bit 14 **PLOOPBK**：PHY 环回位
1 = 所有发送的数据都返回到 MAC。双绞线接口被禁止。
0 = 正常操作

bit 13-12 未用：读为 0

bit 11 **PPWRSV**：PHY 掉电位
1 = PHY 关闭
0 = 正常操作

bit 10 保留：保持为 0

bit 9 未用：读为 0

bit 8 **PDPXMD**：PHY 双工模式位
1 = MAC 工作于全双工模式
0 = MAC 工作于半双工模式

bit 7 保留：保持为 0

bit 6-0 未用：读为 0

寄存器 18-11： PHSTAT1：物理层状态寄存器 1

U-0	U-0	U-0	R-1	R-1	U-0	U-0	U-0
—	—	—	PHFDPX	PHHDPX	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/LL-0	R/LH-0	U-0
—	—	—	—	—	LLSTAT	JBSTAT	—
bit 7							bit 0

图注：	1 = 置 1		
R = 只读位	0 = 清零	U = 未用位，读为 0	
-n = POR 值	R/L = 只读锁存位	LL = 低电平锁存位	LH = 高电平锁存位

bit 15-13 未用：读为 0

bit 12 **PHFDPX**：PHY 全双工功能位
1 = 在全双工模式下， PHY 工作速率可达 10 Mbps （该位始终置 1）

bit 11 **PHHDPX**：PHY 半双工功能位
1 = 在半双工模式下， PHY 工作速率可达 10 Mbps （该位始终置 1）

bit 10-3 未用：读为 0

bit 2 **LLSTAT**：PHY 锁存链路状态位
1 = 链路建立，并且自上次读 PHSTAT1 之后一直处于建立状态
0 = 链路断开，或上次读 PHSTAT1 之后断开一段时间

bit 1 **JBSTAT**：PHY 锁存超时传输状态位
1 = 上次读 PHSTAT1 之后， PHY 检测到有一次数据发送满足超时传输条件
0 = 上次读 PHSTAT1 之后， PHY 未检测到任何超时传输

bit 0 未用：读为 0

寄存器 18-12： PHCON2：PHY 控制寄存器 2

U-0	R/W-0	R/W-0	R/W-x	R/W-x	R/W-0	R/W-0	R/W-0
—	FRCLNK	TXDIS	r	r	JABBER	r	HDLDIS
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
r	r	r	r	r	r	r	r
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 15 未用：读为 0

bit 14 **FRCLNK**：PHY 强制建立链路位
1 = 即使未检测到链路伙伴也强制建立链路
0 = 正常操作

bit 13 **TXDIS**：双绞线发送器禁止位
1 = 禁止双绞线发送器
0 = 正常操作

bit 12-11 保留：写为 0

bit 10 **JABBER**：超时传输校正禁止位
1 = 禁止超时传输校正
0 = 正常操作

bit 9 保留：写为 0

bit 8 **HDLDIS**：PHY 半双工环回禁止位
当 PHCON1<8> = 1 或 PHCON1<14> = 1 时：
该位被忽略。
当 PHCON1<8> = 0 且 PHCON1<14> = 0 时：
1 = 发送数据仅在双绞线接口送出
0 = 发送数据将环回到 MAC，并在双绞线接口送出

bit 7-0 保留：写为 0

寄存器 18-13： PHSTAT2：物理层状态寄存器 2

U-0	U-0	R-0	R-0	R-0	R-0	R-x	U-0
—	—	TXSTAT	RXSTAT	COLSTAT	LSTAT	DPXSTAT[1]	—
bit 15							bit 8

U-0	U-0	R-0	U-0	U-0	U-0	U-0	U-0
—	—	PLRITY	—	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 15-14 未用：读为 0

bit 13 **TXSTAT**：PHY 发送状态位
1 = PHY 正在发送数据
0 = PHY 不在发送数据

bit 12 **RXSTAT**：PHY 接收状态位
1 = PHY 正在接收数据
0 = PHY 不在接收数据

bit 11 **COLSTAT**：PHY 冲突状态位
1 = 当前发生了冲突
0 = 当前未发生冲突

bit 10 **LSTAT**：PHY 链路状态位 （非锁存位）
1 = 链路建立
0 = 链路断开

bit 9 **DPXSTAT**：PHY 双工状态位 [1]
1 = PHY 配置为全双工操作 （PHCON1<8> 置 1）
0 = PHY 配置为半双工操作 （PHCON1<8> 清零）

bit 8-6 未用：读为 0

bit 5 **PLRITY**：极性状态位
1 = TPIN+/TPIN- 上信号的极性相反
0 = TPIN+/TPIN- 上信号的极性正确

bit 4-0 未用：读为 0

注 1：双工模式 / 状态位的复位值取决于 LED 到 LEDB 引脚的连接 （更多详细信息，请参见**第 18.1.2 节 “LED 配置”**）。

寄存器 18-14： PHLCON：PHY 模块 LED 控制寄存器

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0
r	r	r	r	LACFG3	LACFG2	LACFG1	LACFG0
bit 15							bit 8

R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-x
LBCFG3	LBCFG2	LBCFG1	LBCFG0	LFRQ1	LFRQ0	STRCH	r
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 15-14 **保留：**写为 0

bit 13-12 **保留：**写为 1

bit 11-8 **LACFG3:LACFG0：**LEDA 配置位
0000 = 保留
0001 = 显示发送活动 （时间可延长）
0010 = 显示接收活动 （时间可延长）
0011 = 显示冲突活动 （时间可延长）
0100 = 显示链路状态
0101 = 显示双工状态
0110 = 保留
0111 = 显示发送和接收活动 （时间可延长）
1000 = 亮
1001 = 灭
1010 = 快速闪烁
1011 = 慢速闪烁
1100 = 显示链路状态和接收活动 （时间总是延长的）
1101 = 显示链路状态和发送 / 接收活动 （时间总是延长的）
1110 = 显示双工状态和冲突活动 （时间总是延长的）
1111 = 保留

bit 7-4 **LBCFG3:LBCFG0：**LEDB 配置位
0000 = 保留
0001 = 显示发送活动 （时间可延长）
0010 = 显示接收活动 （时间可延长）
0011 = 显示冲突活动 （时间可延长）
0100 = 显示链路状态
0101 = 显示双工状态
0110 = 保留
0111 = 显示发送和接收活动 （时间可延长）
1000 = 亮
1001 = 灭
1010 = 快速闪烁
1011 = 慢速闪烁
1100 = 显示链路状态和接收活动 （时间总是延长的）
1101 = 显示链路状态和发送 / 接收活动 （时间总是延长的）
1110 = 显示双工状态和冲突活动 （时间总是延长的）
1111 = 保留

bit 3-2 **LFRQ1:LFRQ0：**LED 脉冲延长时间配置位 （见表 18-1）
11 = 保留
10 = LED 事件延长 TLSTRCH
01 = LED 事件延长 TMSTRCH
00 = LED 事件延长 TNSTRCH

bit 1 **STRCH：**LED 脉冲延长使能位
1 = 根据 LFRQ1:LFRQ0 的配置，对可延长 LED 事件进行 LED 脉冲延长
0 = 仅当可延长 LED 事件发生时才显示

bit 0 **保留：**不能修改

18.3 以太网中断

以太网模块可生成多个中断条件。为了适应所有这些中断源，模块具有其自身的中断逻辑结构，与单片机的中断逻辑结构很相似。模块使用了独立的寄存器组来允许和标志不同的中断条件。

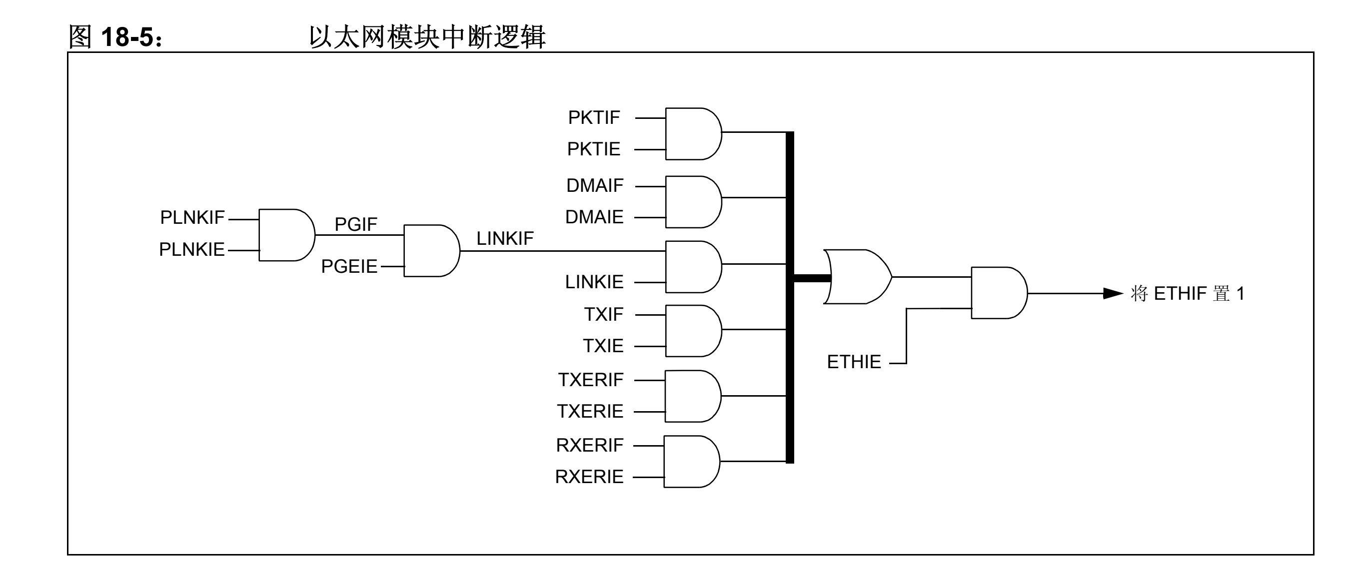
EIE 寄存器包含各个中断源的中断允许位，而 EIR 寄存器包含相应的中断标志位。当发生中断时，中断标志位置1。如果EIE寄存器中允许了中断，并且相应的ETHIE 全局中断允许位置 1，那么单片机主以太网中断标志（ETHIF）也将置 1 （见图 18-5)。

注：	当产生中断条件时，不论相应的中断允许位或相关的全局中断允许位的状态如何，除了 LINKIF 中断标志之外，中断标志位都将置 1。用户软件应在允许中断前，确保先将相应的中断标志位清零。中断标志位用于软件查询。

18.3.1 控制中断 （ETHIE）

寄存器 18-15 至寄存器 18-18 给出了与中断控制相关的 4 个寄存器。

图 18-5： 以太网模块中断逻辑



寄存器 18-15： **EIE**：以太网中断允许寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 未用：读为 0

bit 6 **PKTIE**：接收数据包等待处理中断允许位
1 = 允许接收数据包等待处理中断
0 = 禁止接收数据包等待处理中断

bit 5 **DMAIE**：DMA 中断允许位
1 = 允许 DMA 中断
0 = 禁止 DMA 中断

bit 4 **LINKIE**：链路状态改变中断允许位
1 = 允许来自 PHY 的链路状态改变中断
0 = 禁止链路改变中断

bit 3 **TXIE**：发送使能位
1 = 允许发送中断
0 = 禁止发送中断

bit 2 未用：读为 0

bit 1 **TXERIE**：发送错误中断允许位
1 = 允许发送错误中断
0 = 禁止发送错误中断

bit 0 **RXERIE**：接收错误中断允许位
1 = 允许接收错误中断
0 = 禁止接收错误中断

寄存器 18-16： EIR：以太网中断请求 （标志）寄存器

U-0	R-0	R/C-0	R-0	R/C-0	U-0	R/C-0	R/C-0
—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF
bit 7							bit 0

图注：			
R = 可读位	C = 可清零位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 未用：读为 0

bit 6 **PKTIF**：接收数据包等待处理中断标志位
1 = 接收缓冲区中有一个或多个未处理的数据包；当 PKTDEC 置 1 时 PKTIF 清零。
0 = 接收缓冲区为空

bit 5 **DMAIF**：DMA 中断标志位
1 = DMA 复制或校验和计算已完成
0 = 没有等待响应的 DMA 中断

bit 4 **LINKIF**：链路改变中断标志位
1 = PHY 报告链路状态已经改变，读 PHIR 寄存器清零该位
0 = 链路状态未改变

bit 3 **TXIF**：发送中断标志位
1 = 发送请求已结束
0 = 没有等待响应的发送中断

bit 2 未用：读为 0

bit 1 **TXERIF**：发送错误中断标志位
1 = 发生了发送错误
0 = 未发生发送错误

bit 0 **RXERIF**：接收错误中断标志位
1 = 由于缓冲区空间不足或者数据包数已经达到 255，而中止数据包接收
0 = 没有等待响应的接收错误中断

寄存器 18-17： PHIE：PHY 中断允许寄存器

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
r	r	r	r	r	r	r	r
bit 15							bit 8

R/W-x	R/W-x	R/W-0	R/W-0	R/W-x	R/W-x	R/W-0	R/W-0
r	r	r	PLNKIE	r	r	PGEIE	r
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 15-6 **保留**：写为 0，读取则忽略

bit 5 **保留**：保持为 0

bit 4 **PLNKIE**：PHY 链路改变中断允许位
1 = 允许 PHY 链路改变中断
0 = 禁止 PHY 链路改变中断

bit 3-2 **保留**：写为 0，读取则忽略

bit 1 **PGEIE**：PHY 全局中断允许位
1 = 允许 PHY 中断
0 = 禁止 PHY 中断

bit 0 **保留**：保持为 0

寄存器 18-18： PHIR：PHY 中断请求 （标志）寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
r	r	r	r	r	r	r	r
bit 15							bit 8

R-x	R-x	R-0	R/SC-0	R-0	R/SC-0	R-x	R-0
r	r	r	PLNKIF	r	PGIF	r	r
bit 7							bit 0

图注：			
R = 可读位	SC = 自清零位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 15-6 **保留**：不能修改

bit 5 **保留**：读为 0

bit 4 **PLNKIF**：PHY 链路改变中断标志位
1 = 上次读 PHIR 之后 PHY 链路状态已经改变，读取时复位为 0
0 = 上次读 PHIR 之后 PHY 链路状态未改变

bit 3 **保留**：读为 0

bit 2 **PGIF**：PHY 全局中断标志位
1 = 上次读 PHIR 之后，发生了一个或多个被允许的 PHY 中断，读取时复位为 0
0 = 未发生 PHY 中断

bit 1 **保留**：不能修改

bit 0 **保留**：读为 0

18.3.1.1 接收错误中断 （RXERIF）

接收错误中断用来指示接收缓冲区的溢出状况。或者，该中断也可能指示接收缓冲区中数据包太多，再接收的话将造成 EPKTCNT 寄存器溢出。

如果在接收数据包时，接收缓冲区空间完全用尽，或者 EPKTCNT 达到 255 已不能再增加，则中止正在接收的数据包（永久丢失），并将 RXERIF 位置 1。一旦置 1，RXERIF 就只能由固件或复位条件清零。如果允许接收错误中断和以太网中断 （RXERIE 和 ETHIE 均置 1），则将产生以太网中断。如果未允许接收错误中断（RXERIE 或 ETHIE 清零），用户应用程序可以查询 RXERIF 并执行适当的操作。

通常，在出现接收错误时，应用程序将处理接收缓冲区中等待处理的数据包，然后通过递增 ERXRDPT 寄存器（首先是低字节）和递减 EPKTCNT 寄存器，为以后接收的数据包留出更多空间。关于数据包处理的更多信息，请参见**第 18.5.3.3 节 “释放接收缓冲区空间”**。处理之后，应用程序应当立即清零 RXERIF 位。

18.3.1.2 发送错误中断 （TXERIF）

发送错误中断用来指示发送被中止的情况。中止可由以下任何条件造成：

1. 发生过多的冲突，超过 MACLCON1 寄存器中重发最大数 （RETMAX）位定义的数目。
2. 发生了MACLCON2寄存器中冲突窗口（COLWIN）位定义的迟冲突。
3. 发送 64 字节之后发生了冲突（LATECOL 置 1）。
4. 由于介质被持续占用的时间过长，达到了延迟时限，无法发送数据包。DEFER 位（MACON4<6>）清零。
5. 在未将 HFRMEN 位 （MACON3<2>）或每个数据包的 POVERRIDE 和 PHUGEEN 位置 1 的情况下，尝试发送大于 MAMXFL 寄存器中定义的最大帧长的数据包。

在以上任何情况下， TXERIF 标志位都将置 1。一旦置 1，就只能由固件或复位条件清零。如果允许了发送错误中断 （TXERIE 和 ETHIE 均置 1），则将产生以太网中断。如果未允许发送错误中断（TXERIE 或 ETHIE 清零），应用程序可以查询 TXERIF 并执行适当的操作。处理中断之后，应当立即清零标志位。

在发送中止之后， TXRTS 位清零、TXABRT 位（ESTAT<1>）置 1、发送状态向量将写入 ETXND + 1 位置处。MAC 不会自动尝试重发数据包。应用程序可以读发送状态向量和 LATECOL 位来判断中止的原因。确定问题和解决方法之后，应用程序应清零 LATECOL 位（如果置 1）和 TXABRT 位，以便可以准确地检测到以后发生的中止。

在全双工模式下，只有条件 5 可以引发中断。在全双工网络中，不会发生冲突及其他与网络共享相关的问题。导致发送错误中断的条件满足发送中断的要求。因而，当发生该中断时， TXIF 也将同时置 1。

18.3.1.3 发送中断 （TXIF）

发送中断用于指示所请求的数据包发送已经结束（TXRTS 位由 1 转变为 0）。在发送完成、中止或应用程序取消发送时，TXIF 标志都会置 1。如果应用程序未清零 TXRTS 位，而 TXABRT 位未置 1，则说明数据包发送成功。一旦 TXIF 置 1，就只能由固件或复位条件清零。如果允许了发送中断 （TXIE 和 ETHIE 均置 1），则将产生中断。如果未允许发送中断 （TXIE 或 ETHIE 清零），应用程序可以查询 TXIF 位并执行适当的操作。

18.3.1.4 链路改变中断 （LINKIF）

LINKIF 用于指示链路状态发生了改变。当前的链路状态可通过 LLSTAT（PHSTAT1<2>）或 LSTAT（PHSTAT2<10>）位获得（见寄存器 18-11 和寄存器 18-13）。与其他中断源不同，链路状态改变中断产生于 PHY 模块的内部；必须采取另外的步骤来允许该中断。

在默认复位状态下，无论如何都不会将 LINKIF 位置 1。要允许该中断，必须将 PLNKIE 和 PGEIE 位都置 1。当允许该中断时，LINKIF 位将覆盖 PGIF 位的内容。由于 PHY 仅支持一个中断，因此 PGIF 位始终和 PLNKIF 位的状态保持一致 （两个 PHY 允许位都置 1 时）。

一旦 LINKIF 置 1，就只能由软件或复位清零。如果允许了链路改变中断（LINKIE、PLNKIE、PGEIE 和 ETHIE 均置 1），则将产生中断。如果未允许链路改变中断（LINKIE、PLNKIE、PGEIE 或 ETHIE 清零），用户应用程序可以查询 PLNKIF 并执行适当的操作。

LINKIF 是只读位。由于读 PHY寄存器需要一段不可忽略的时间，应用程序可以将 PLNKIE 和 PGEIE 置 1，然后查询 LINKIF 标志位。对 PHIR 寄存器执行 MII 读操作将自动清零 LINKIF、 PGIF 和 PLNKIF 位，并允许以后的链路改变中断。关于访问 PHY 寄存器的更多信息，请参见**第 18.2.5 节 “PHY 寄存器”**。

18.3.1.5 DMA 中断 （DMAIF）

DMA 中断指示 DMA 模块已经完成存储器内容复制或校验和计算 （DMAST 位从 1 转换为 0）。此外，如果应用程序通过手动清零 DMAST 位而取消 DMA 操作，也会引发该中断。DMAIF 一旦置 1，就只能由固件或复位条件清零。如果允许了 DMA 中断，则将产生以太网中断。如果未允许 DMA 中断，用户应用程序可以查询 DMAIF 标志位的状态，并执行适当的操作。处理中断之后，应当立即清零标志位。

18.3.1.6 接收数据包等待处理中断 （PKTIF）

接收数据包等待处理中断用来指出接收缓冲区中存在一个或多个数据包，并用来通知接收到新的数据包。当接收缓冲区中有至少一个数据包时， PKTIF 标志位置 1。换言之，只要以太网数据包计数寄存器 （EPKTCNT）非零，该中断标志都将置 1。

当允许接收数据包等待处理中断（PKTIE 和 INTIE 均置 1）时，只要成功接收到新的数据包并写入接收缓冲区，就会产生以太网中断。如果未允许接收数据包等待处理中断（PKTIE 或 INTIE 清零），用户应用程序可以查询 PKTIF 位并执行适当的操作。

PKTIF 位只能通过在软件中将 EPKTCNT 寄存器减至 0 来间接清零，或者通过复位条件清零。关于清零 EPKTCNT 寄存器的更多信息，请参见**第 18.5.3 节 “接收数据包”**。当处理完接收缓冲区中的最后一个数据包时， EPKTCNT 变为 0， PKTIF 位自动清零。

 超前信息

18.4 模块初始化

只有在对器件的某些设置进行初始化之后，以太网模块才能发送和接收数据包。根据应用的不同，一些配置选项可能需要更改。正常情况下，这些在复位之后一次性配置即可，之后就不再变动。

18.4.1 接收缓冲区

接收缓冲区必须通过设置 ERXST 和 ERXND 指针进行初始化，然后才能接收数据包。 ERXST 和 ERXND 地址（含）之间的所有存储空间都专供接收硬件使用。建议将 ERXST 指针设定为偶地址。

如果应用程序要处理大量的数据，并频繁传送数据包，可以将大部分存储空间分配作为接收缓冲区。如果应用程序需要保存旧的数据包，或者准备同时发送几个数据包，则应分配较少的接收存储器空间。

当设定 ERXST 或 ERXND 指针时，ERXWRPT 指针寄存器将自动更新为 ERXST 中的值。当硬件开始写接收到的数据时，将使用ERXWRPT 中的地址作为写操作的起始地址。为了进行跟踪，ERXRDPT 寄存器应设定为相同的值。在设定ERXRDPT寄存器时，应首先写ERXRDPTL，然后再写 ERXRDPTH。更多信息，请参见**第 18.5.3.3 节“释放接收缓冲区空间”**。

18.4.2 发送缓冲区

所有未用于接收缓冲区的存储器空间都属于发送缓冲区。要发送的数据应写入未用的空间。不过，在发送数据包之后，硬件会在数据包的最后一个字节之后，向存储器写入一个 7 字节的状态向量。因此，应用程序应在数据包之间，以及接收缓冲区的起始单元之前，保留至少 7 字节的空间。

18.4.3 接收过滤器

通过写 ERXFCON 寄存器可以使能或禁止接收过滤器。关于如何配置接收过滤器的信息，请参见**第 18.8 节“接收过滤器”**。

18.4.4 等待 PHY 起振定时器

如果在使能模块之后立即执行初始化过程（ECON2<5> 置 1），则应查询 PHYRDY 位，确保修改 PHY 寄存器之前已经经过足够的时间（典型值为 1 ms）。关于 PHY 起振定时器的更多信息，请参见**第 18.1.3.1 节“起振定时器”**。

18.4.5 MAC 初始化设置

初始化期间需要对几个 MAC 寄存器进行配置。这只需要在初始化期间进行一次即可，设定次序无关紧要。

1. 将 MARXEN 位（MACON1<0>）置 1 以使能 MAC 接收帧。如果使用全双工模式，大部分应用还应该将 TXPAUS 和 RXPAUS 置 1，以使 IEEE 定义的流量控制生效。
2. 配置 MACON3 寄存器中的 PADCFG<2:0>、TXCRCEN 和 FULDPX 位。大部分应用应当使能自动填充至 60 字节，并始终追加一个有效的 CRC。为了方便起见，许多应用可能还需要将 FRMLNEN 位置 1，以使能帧长状态报告。如果应用要连接到配置为全双工模式的远程节点，则应将 FULDPX 位置 1 ；否则，应保持清零。
3. 配置 MACON4 中的位。为了符合 IEEE 802.3 的规定，应确保将DEFER位（MACON4<6>）置1。
4. 将 MAMXFL 寄存器设定为允许接收或发送的最大帧长。网络节点通常设计为处理不大于 1518 字节的数据包。
5. 配置背对背包间间隔寄存器 MABBIPG。当使用全双工模式时，大部分应用将该寄存器设定为 15h ；当使用半双工模式时，设定为 12h。关于配置包间间隔的更多详细描述，请参见寄存器 18-19。
6. 配置非背对背包间间隔低字节寄存器 MAIPGL。大部分应用将该寄存器设定为 12h。
7. 如果使用半双工模式，应对非背对背包间间隔高字节寄存器 MAIPGH 进行设定。大部分应用将该寄存器设定为 0Ch。
8. 如果使用半双工模式，需要对重发最大数和冲突窗口寄存器 MACLCON1 和 MACLCON2 进行设定。大部分应用不需要修改默认的复位值。在网路线特别长的情况下，可能需要增大 MACLCON2 的默认值。
9. 将本地 MAC 地址写入 MAADR1:MAADR6 寄存器。

寄存器 18-19： **MABBIPG：MAC 背对背包间间隔寄存器**

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 未用：读为 0

bit 6-0 **BBIPG6:BBIPG0：**背对背包间间隔延时时间位

当 FULDPX （MACON3<0>） = 1 时：
在背对背序列中，前一次发送结束和下一次发送起始之间延时半字节的时间。应使用半字节时间的整数倍时间减 3 来对寄存器值进行设定。建议将该寄存器设置为 15h，它表示 IEEE 中规定的最小包间间隔 （IPG） 9.6 μs。

当 FULDPX （MACON3<0>） = 0 时：
在背对背序列中，前一次发送结束和下一次发送起始之间延时半字节的时间。应使用半字节时间的整数倍时间减 6 来对寄存器的值进行设定。建议将该寄存器设置为 12h，它表示 IEEE 中指定的最小包间间隔 （IPG） 9.6 μs。

18.4.6 PHY 初始化设置

根据应用的不同，可能还需要对 3 个 PHY 模块寄存器中的位进行配置。

PDPXMD 位 （PHCON1<8>）控制 PHY 半 / 全双工模式的配置。应用程序必须正确地设定该位和 FULDPX 位（MACON3<0>）。

HDLDIS 位 （PHCON2<8>）禁止数据自动环回。该位应始终置 1。

PHY 寄存器 PHLCON （寄存器 18-14）控制 LEDA 和 LEDB 的输出。如果应用需要不同于默认的 LED 配置，可以修改该寄存器来满足新的要求。在**第 18.1.2 节 “LED 配置”**中讨论了 LED 操作的设置。

18.4.7 禁止以太网模块

在某些情况下，可能会有较长时间不会用到以太网模块。例如，有些情况只需要在发生特殊事件时才发送或接收以太网数据包。在这些情况下，可以选择性地关闭模块。

要选择性地禁止模块：

1. 通过清零 RXEN 位禁止数据包接收。
2. 通过查询 RXBUSY 位 （ESTAT<2>）等待正在接收中的数据包完成接收。应先将该位清零。
3. 通过确认 TXRTS 位（ECON1<3>）是否清零等待当前所有的发送结束。
4. 清零 ETHERN 位。这使所有 MAC、MII 和 PHY 寄存器变成不可访问，并自动清零 PHYRDY 位。

18.5 发送和接收数据

以太网协议（IEEE 标准 802.3）非常详细地描述了基于帧的 10 Mbps 串行通信系统。在讨论以太网模块的实际使用之前，有必要简要概述一下典型以太网数据帧的结构。以下内容假设用户已经对 IEEE 802.3 有一定程度的了解。对于那些需要更多信息的用户，可以参考官方标准或其他关于以太网的参考文章，以获取更全面的信息。

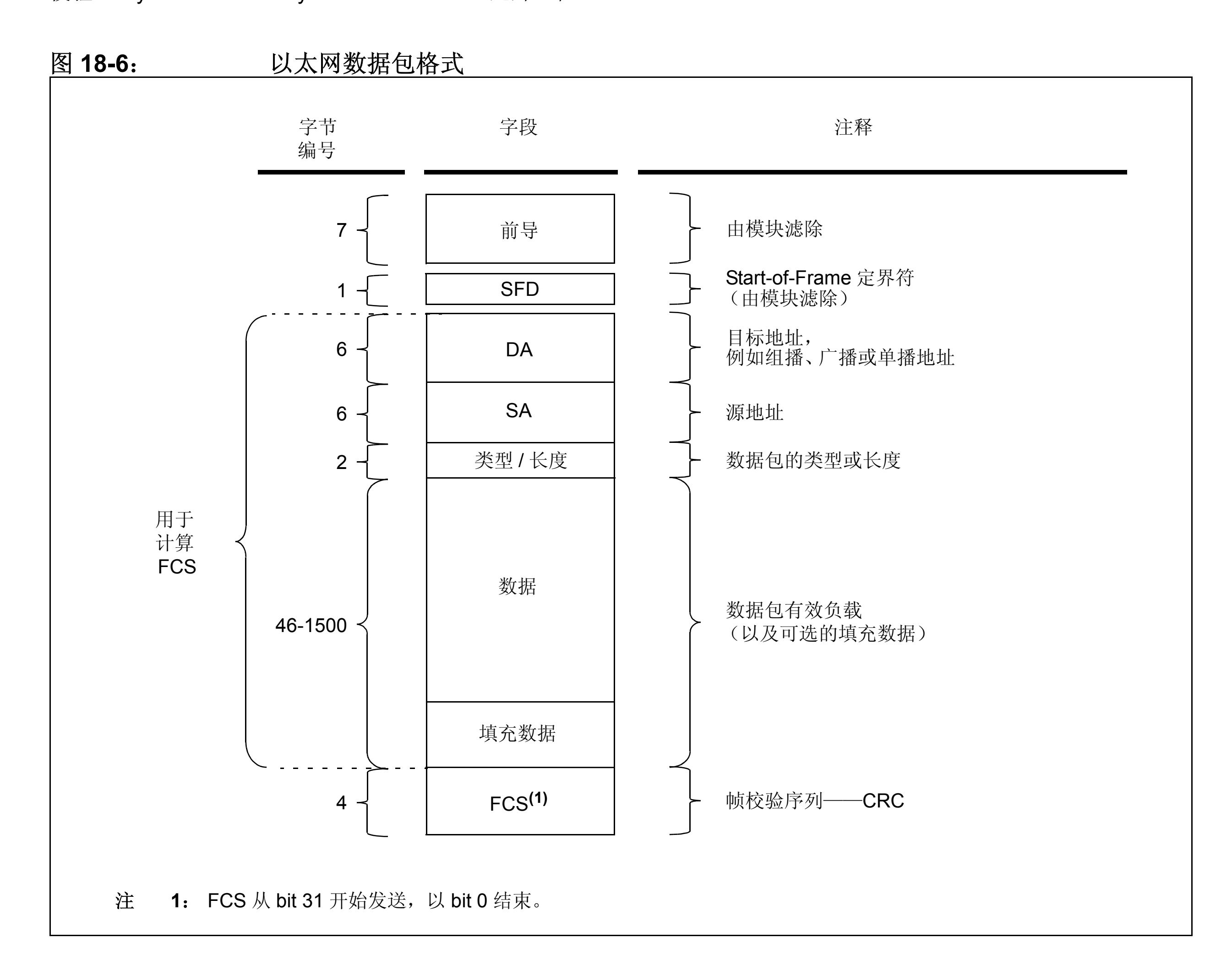
18.5.1 数据包格式

符合 IEEE 802.3 标准的正常以太网帧的长度介于 64 和 1518 字节之间。它们由 5 或 6 个不同的字段组成。这些字段分别是：目标 MAC 地址、源 MAC 地址、类型 / 长度字段、数据有效负载、可选的填充字段和循环冗余校验（Cyclic Redundancy Check， CRC）。此外，在以太网介质上进行发送时，在以太网数据包的前面还要添加一个 7 字节的前导字段和一个 Start-of-Frame 定界符字节。因此，双绞线电缆上的通信数据将会如图 18-6 所示。

18.5.1.1 前导 /Start-of-Frame 定界符

当以太网模块发送和接收数据时，将在发送时自动生成前导和 Start-of-Frame 定界符字节，而在接收时从数据包中剥离它们。还可以在发送时自动地根据需要生成 CRC 字段和填充数据，并在接收时校验 CRC 数据。用户应用程序无需创建或处理这些字段，或手动校验CRC 数据。不过，当数据包到达时，填充数据和 CRC 字段会被写入接收缓冲区，所以用户应用程序可以根据需要对其进行评估。

图 **18-6**： 以太网数据包格式



18.5.1.2 目标地址

目标地址字段为 6 字节长，包含数据包要发往的设备的 MAC 地址。如果 MAC 地址首字节的最低有效位置 1，则地址为组播目标地址。例如， 01-00-00-00-F0-00 和 33-45-67-89-AB-CD 是组播地址，而 00-00-00-00-F0-00 和 32-45-67-89-AB-CD 不是组播地址。

带组播目标地址的数据包将被送至一组选定的以太网节点。如果目标地址字段是保留的组播地址 FF-FF-FF-FF-FF-FF，则数据包是广播数据包，它将被发送给共享网络的每一个节点。如果 MAC 地址首字节的最低有效位清零，则地址为单播地址，只能供被寻址的节点使用。

以太网模块在内部实现了一些接收过滤器，它们可以用来滤除或接收具有组播、广播和 / 或单播目标地址的数据包。在发送数据包时，应用程序需要负责将所需的目标地址写入发送缓冲区。

18.5.1.3 源地址

源地址字段为 6 字节长，包含创建以太网数据包的节点的 MAC 地址。以太网模块用户必须为所使用的每个单片机生成一个唯一的 MAC 地址。

MAC地址包含两个部分。前三个字节是组织唯一标识符（OUI）。OUI 由 IEEE 分配。后三个字节是由购买 OUI 的公司定义的地址字节。

在发送数据包时，应用程序必须将分配的源 MAC 地址写入发送缓冲区。模块不会自动发送 MAADR 寄存器的内容，这些寄存器是供单播接收过滤器使用的。

18.5.1.4 类型 / 长度

类型 / 长度字段是一个 2 字节字段，它定义后面数据包中的数据属于何种协议。或者，如果该字段填充的数值小于等于 05DCh （1500），则字段视为长度字段，它指定后面的数据字段中非填充数据的数量。使用专用网络的用户可以选择将此字段用作长度字段；使用诸如因特网协议（Internet Protocol，IP）或地址解析协议（Address Resolution Protocol，ARP）的用户，在发送数据包时应将该字段设定为协议规范中所定义的适当类型。

18.5.1.5 数据

数据字段是一个可变长度的字段，长度可以为 0 到 1500 字节。更大的数据包会违反以太网标准，会被大部分以太网节点抛弃。不过，在超大帧使能位 HFRMEN 置 1 时（MACON3<2> = 1），以太网模块可以发送和接收更大的数据包。

18.5.1.6 填充数据

填充字段也是一个可变长度的字段，添加该字段是为了在数据有效负载较小时也能满足 IEEE 802.3 规范要求。以太网数据包的目标、源、类型、数据和填充字段合起来不能小于 60 字节。加上必要的 4 字节 CRC 字段，数据包不能小于 64 字节。如果数据字段长度小于 46 字节，则需要填充字段。

在发送数据包时，如果 PADCFG2:PADCFG0 位（MACON3<7:5>）配置为执行填充操作，以太网模块会自动生成填充数据 0。否则，用户应用程序需要在发送数据包之前，为其添加填充数据。如果应用程序命令发送小于规定大小的数据包，模块并不会拒绝该操作。

在接收数据包时，模块会自动拒收长度小于 18 字节的数据包。所有 18 字节或更大的数据包将按标准接收过滤条件进行处理，可能会作为正常通信数据而接收。因为模块仅拒绝长度小于 18 字节的数据包，固件应对每个接收到的数据包进行长度检查并拒绝长度小于 64 字节的数据包，以符合 IEEE 802.3 规范要求，这一点很重要。

18.5.1.7 CRC

CRC 字段是一个 4 字节的字段，其中包含工业标准的 32 位 CRC 值，该值是从目标、源、类型、数据和填充字段计算得到的。

在接收数据包时，以太网模块会检查每个到达数据包的 CRC。如果 CRCEN 位置 1，将自动丢弃具有无效 CRC 的数据包。如果 CRCEN 清零，并且数据包满足其他所有接收过滤条件，则将数据包写入接收缓冲区，应用程序可以通过读接收状态向量来确定 CRC 是否有效（见**第 18.5.3 节 “接收数据包”**）。

在发送数据包时，模块会自动生成一个有效的 CRC，并自动发送它（如果 PADCFG2:PADCFG0 位配置为执行此操作）。否则，必须由用户应用程序生成 CRC，并将其写入发送缓冲区。鉴于 CRC 计算的复杂性，强烈建议用户让模块自动计算和添加 CRC。

18.5.2 发送数据包

在发送时，以太网模块的 MAC 会自动生成前导字段和 Start-of-Frame 定界符字段。此外，MAC 可以生成任何填充数据（如果需要）和 CRC（如果配置为这样做）。其他所有帧字段必须由应用程序生成，并将其写入发送缓冲存储区。

此外，以太网模块要求每个发送数据包之前有一个数据包控制字节。控制字节的结构如图 18-7 所示。在发送数据包之前，应对修改发送特性的MAC寄存器进行初始化设置（请参见**第 18.4 节 “模块初始化”**）。

图 18-7： 数据包控制字节的格式

—	—	—	—	PHUGEEN	PPADN	PCRCEN	POVERRIDE
bit 7							bit 0

bit 7-4　未使用

bit 3　**PHUGEEN：**数据包超大帧使能位

<u>当 POVERRIDE = 1 时：</u>
1 = 发送完整的数据包
0 = MAC 发送 MAMXFL 规定的最大字节数。如果数据包大于 MAMXFL，发送字节数达到 MAMXFL 后将中止发送。

<u>当 POVERRIDE = 0 时：</u>
该位被忽略。

bit 2　**PPADN：**数据包填充使能位

<u>当 POVERRIDE = 1 时：</u>
1 = 如果数据包小于 60 字节，将用 0 填充至 60 字节
0 = 直接发送数据包，不添加任何填充字节

<u>当 POVERRIDE = 0 时：</u>
该位被忽略。

bit 1　**PCRCEN：**数据包 CRC 使能位

<u>当 POVERRIDE = 1 时：</u>
1 = 计算出一个有效的 CRC，并加入帧内
0 = 不追加 CRC。将帧的最后 4 个字节作为 CRC 来验证其有效性。

<u>当 POVERRIDE = 0 时：</u>
该位被忽略。

bit 0　**POVERRIDE：**数据包改写位

1 = PCRCEN、 PPADN 和 PHUGEEN 的值将改写 MACON3 定义的配置
0 = 使用 MACON3 中的值确定如何发送数据包

图 18-8 给出了完整的发送数据包在存储器中的结构示例。以如下方式构建和发送数据包：

1. 设置ETXST指针使其指向缓冲区中的未用单元。这将是数据包控制字节的存储位置。在示例中，其地址为 0120h。建议该地址使用偶地址。
2. 将数据包控制字节的数据、目标地址、源 MAC 地址、类型 / 长度和数据有效负载顺序地写入以太网缓冲区。
3. 将 ETXND 指针设置为指向数据有效负载的末字节。在示例中，它将设定为 0156h。
4. 完成以上操作后，清零 TXIF 标志位（EIR<3>），并将 TXIE 位 （EIE<3>）和 ETHIE 位置 1，以允许发生中断 （如果需要）。
5. 通过将 TXRTS 位（ECON1<3>）置 1 来启动发送过程。

如果将 TXRTS 位置 1 时正在进行 DMA 操作，那么模块将等待 DMA 操作完成，之后再尝试发送数据包。这个延时是必要的，因为 DMA 和发送引擎共用同一个存储器访问端口。类似地，如果在 TXRTS 已经置 1 之后将 DMAST 位置 1，DMA 将等待 TXRTS 位清零，然后再执行操作。

当正在进行发送时，不应修改 ETXST 和 ETXND 指针。如果需要取消发送，可以清零 TXRTS 位。

当数据包发送结束，或由于错误 / 取消而中止时，将会产生以下情况：

- TXRTS 位清零
- 在ETXND + 1所指向的缓冲区位置写入一个7字节的发送状态向量。
- TXIF 标志位置 1
- 将产生中断 （如果允许）
- ETXST 与 ETXND 指针不改变。

要检查数据包是否成功发送，可以读取 TXABRT 位。如果它已经置 1，可以查询 LATECOL 位及发送状态向量中的各个字段来确定原因。发送状态向量的结构如表 18-3 所示。多字节字段使用小尾数法写入。

图 18-8： 发送数据包结构示例

缓冲区指针	地址	存储器		说明
ETXST = 0120h	0120h	0Eh	控制	PHUGEEN、PPADN、PCRCEN 和 POVERRIDE
	0121h	data[1]	数据包	目标地址、源地址和类型 / 长度和数据
	0122h	data[2]		
		⋮		
ETXND = 0156h	0156h	data[m]		
	0157h	tsv[7:0]	状态向量	由硬件写入状态向量
	0158h	tsv[15:8]		
	0159h	tsv[23:16]		
	016Ah	tsv[31:24]		
	016Bh	tsv[39:32]		
	016Ch	tsv[47:40]		
	016Dh	tsv[55:48]		
	016Eh			下一数据包开头

表 **18-3**： 发送状态向量

位	字段	说明
55-52	零	0
51	发送 VLAN 标记帧	帧的长度 / 类型字段包含的值为 VLAN 协议标识符 8100h。
50	应用背压	已应用载波侦听式背压。
49	发送暂停控制帧	发送帧是一个含有有效暂停操作码的控制帧。
48	发送控制帧	发送帧是一个控制帧。
47-32	线上发送的总字节数	为当前数据包在线上发送的总字节数，包括所有因冲突尝试的字节。
31	发送数据不足	保留。该位始终为 0。
30	发送超大帧	帧字节数大于 MAXXFL。
29	发送迟冲突	冲突发生在冲突窗口 （MACLCON2）之外。
28	发送冲突过量	数据包在冲突数超过重发最大数 （MACLCON1）后中止发送。
27	发送延时过长	数据包延时时间超过 24,287 个比特时间 （2.4287ms）。
26	发送数据包延时	数据包至少延迟了一次，但延时未超出规定值。
25	发送广播	数据包的目标地址是广播地址。
24	发送组播	数据包的目标地址是组播地址。
23	发送完成	数据包发送已经完成。
22	发送长度超出范围	指示帧的类型 / 长度字段大于 1500 字节 （类型字段）。
21	发送长度校验错误	指示数据包中帧长度字段的值与实际数据字节长度不符，并且不是类型字段。 MACON3.FRMLNEN 置 1 才能捕捉此错误。
20	发送 CRC 错误	数据包中的 CRC 与内部生成的 CRC 不匹配。
19-16	发送冲突数	在尝试发送期间，当前数据包引起的冲突次数。它适用于成功发送的数据包，因此不会显示可能的最大冲突次数 16。
15-0	发送字节数	帧中的总字节数，不包括冲突字节。

18.5.3 接收数据包

假设接收缓冲区已经初始化，MAC 已经正确配置，并且已经配置了接收过滤器，那么应用程序应执行以下步骤来接收以太网数据包：

1. 将 PKTIE 和 ETHIE 位置 1，以便在接收到数据包时产生以太网中断（如果需要）。
2. 清零 RXERIF 标志，并将 RXERIE 和 ETHIE 都置 1，以便在由于缓冲区空间不足而导致数据包丢失时，产生中断（如果需要）。
3. 通过将RXEN位（ECON1<2>）置1来使能接收。

将 RXEN 置 1 之后，双工模式和接收缓冲区起始与结束指针应该保持不变。此外，为防止意外数据包进入，建议在改变接收过滤器配置（ERXFCON）和 MAC 地址之前，先清零 RXEN 位。

在使能接收之后，未过滤掉的数据包将写入循环接收缓冲区。不符合必要过滤器条件的所有数据包都会被丢弃，应用程序无法通过任何方式确定是否丢弃了数据包。在接收并将数据包成功写入缓冲区之后，会发生如下情况：

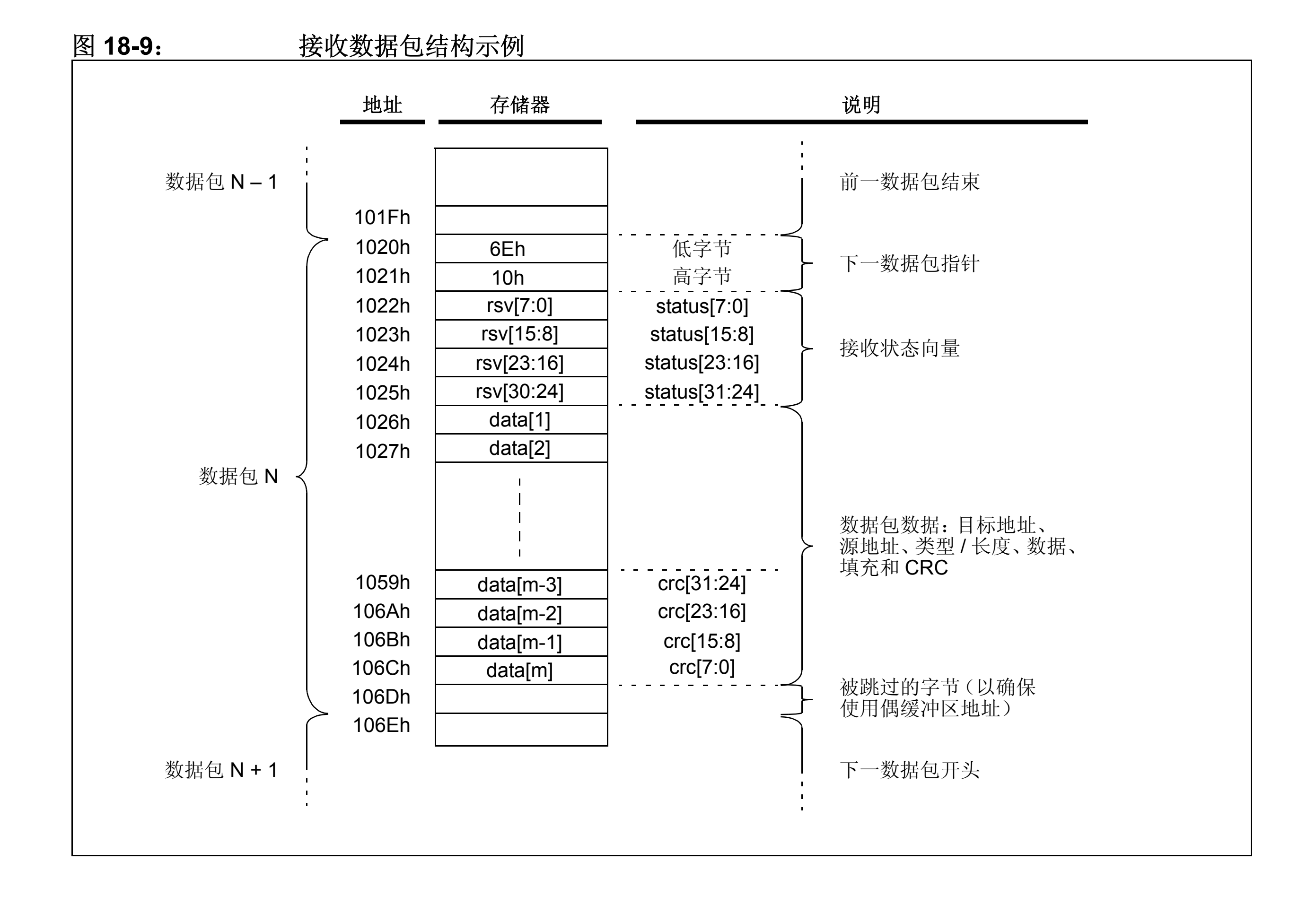
- EPKTCNT 寄存器递增；
- PKTIF 位置 1 ；
- 产生中断（如果允许），以及
- 硬件写指针 ERXWRPT 自动递增。

18.5.3.1 接收数据包结构

图 18-9 显示了接收数据包的结构。数据包的开始是一个 6 字节的报头，报头中含有下一数据包指针，以及接收状态向量（其中含有包括数据包大小在内的接收统计信息）。接收状态报头如表 18-4 所示。

如果数据包中的末字节以奇地址结尾，那么硬件会在递增硬件写指针时，自动追加一个填充字节。这样，所有数据包都将起始于偶地址边界。

图 18-9： 接收数据包结构示例



 超前信息

表 18-4： 接收状态向量

位	字段	说明
31	零	0
30	检测到接收 VLAN 类型	确认当前帧为 VLAN 标记帧。
29	接收未知操作码	确认当前帧为控制帧，但它包含未知操作码。
28	接收暂停控制帧	确认当前帧为控制帧，其中包含有效暂停帧操作码和有效目标地址。
27	接收控制帧	确认当前帧为控制帧，因为有效的类型 / 长度字段指明它为控制帧。
26	多余数据	指示在数据包末尾，额外接收到 1 到 7 位数据。多余的位被丢弃。
25	接收广播数据包	指示接收到的数据包具有有效的广播地址。
24	接收组播数据包	指示接收到的数据包具有有效的组播地址。
23	接收正确	指示数据包具有有效的 CRC，并且无符号错误。
22	长度超出范围	指示帧的类型 / 长度字段大于 1500 字节 （类型字段）。
21	长度校验错误	指示数据包中帧长字段的值与实际数据字节长度不符，并指定了一个有效的长度。
20	CRC 错误	指示帧 CRC 字段的值与 MAC 计算的 CRC 值不匹配。
19	保留	
18	检测到载波事件	指示在上次接收之后的某个时间，检测到载波事件。载波事件与该数据包无关。载波事件是指接收通道上与数据包接收尝试操作无关的活动。
17	保留	
16	长事件 / 丢弃事件	指示数据包接收时间超过 50,000 个比特时间，或上次接收之后丢弃了数据包。
15-0	接收字节数	指示接收帧的长度。这包括目标地址、源地址、类型 / 长度、数据、填充和 CRC 字段。该字段以小尾格式存储。

18.5.3.2 读取接收到的数据包

要处理数据包，正常情况下应用程序将从下一数据包指针开始处读起。应用程序将保存下一数据包指针和接收状态向量的所有必要字节，然后开始读取实际的数据包内容。如果 AUTOINC 位置 1，则可以无需修改 ERDPT 寄存器而顺序地读取整个数据包。读指针会在达到循环接收缓冲区的末尾时自动折回到起始处。

在应用程序需要随机访问数据包的情况下，将需要手动计算ERDPT的值；如果数据包存储时刚好跨越ERXND 到 ERXST 的缓冲区边界，要注意不能超过接收缓冲区的边界。也就是说，在给定数据包起始地址和所需偏移量时，应用程序应遵循公式 18-1 中的逻辑关系。

公式 18-1： 随机访问地址计算

如果数据包起始地址 + 偏移量 > ERXND，则
　　ERDPT = 数据包起始地址 + 偏移量 – (ERXND – ERXST + 1)
否则
　　ERDPT = 数据包起始地址 + 偏移量

18.5.3.3 释放接收缓冲区空间

在用户应用程序处理完数据包（或数据包的部分内容），并需要释放已处理数据所占用的缓冲区空间时，它必须递增接收缓冲区读指针 ERXRDPT。模块总是一直写到接收缓冲区读指针所指的存储单元（不包括该单元）。在要覆盖接收缓冲区读指针所指向的存储单元时，将会中止正在处理的数据包，EXERIF 标志位置 1，并会产生中断（如果允许）。按照这种方式，硬件将永远不会覆盖未处理的数据包。通常，ERXRDPT 将递增至下一数据包指针（位于当前数据包的接收状态向量前）所指向的地址值。遵循这样的过程，就不需要进行任何指针计算来处理在循环接收缓冲区的末尾折回到起始处的问题。

接收缓冲区读指针低字节（ERXRDPTL 寄存器）会在内部进行缓冲，以防止在仅更新一个字节时移动指针。要移动 ERXRDPT，应用程序必须首先写 ERXRDPTL。写操作将更新内部缓冲区，但不会影响寄存器。当应用程序写 ERXRDPTH 时，会同时将内部缓冲的低字节装入 ERXRDPTL 寄存器。ERXRDPT 的两个字节可以按任意次序读取。当读 ERXRDPT 字节时，将返回寄存器的实际值。因而，缓冲的低字节是不可读的。

在处理完每个数据包之后，除了递增接收缓冲区读指针之外，用户应用程序还必须将 PKTDEC 位（ECON2<6>）置 1。这将导致 EPKTCNT 寄存器减 1。递减之后，如果 EPKTCNT 为 0，PKTIF 标志位将自动清零。否则，它将保持置 1，表示接收缓冲区中还有其他数据包在等待处理。尝试将 EPKTCNT 减到 0 以下的操作会被忽略。此外，如果 EPKTCNT 寄存器已达到最大值 255，那么将中止接收所有新的数据包，即便是还有可用的缓冲区空间。RXERIF 位将置 1，并将产生中断（如果允许），以此来报告该错误。为防止这种情况发生，每当处理完一个数据包时，用户应用程序必须正确地递减计数器。

由于只有一个指针可用于控制缓冲区的使用权，应用程序必须按照数据包的接收顺序来处理数据包。如果要保存某个数据包以便稍后进行处理，应用程序应将该数据包复制到存储器中的未用位置。通过使用内部集成的 DMA 控制器可以有效地实现该操作（见**第 18.9 节“直接存储器访问控制器”**）。

18.5.3.4 接收缓冲区剩余空间

任何时候如果应用程序需要了解接收缓冲区还剩余多少空间，应该读硬件写指针（ERXWRPT 寄存器），并且将其与 ERXRDPT 寄存器进行比较。结合接收缓冲区的已知大小，就可以算出剩余空间的大小。

注：	只有在成功接收数据包之后，ERXWRPT 寄存器才会更新。如果应用程序在成功接收另一个数据包之前读 ERXWRPT，那么返回的值可能小于实际值，两者的差值为所允许的最大帧长（MAMXFLN）＋ 7。此外，当应用程序读 ERXWRPT 的一个字节时，可能会有新的数据包到达，并在应用程序读 ERXWRPT 的另一字节之前更新了指针。

如果在使能接收硬件时读 ERXWRPT 寄存器，则必须特别小心，确保低字节和高字节是作为一对被读取的。

为确保获得配对的高低字节：

1. 读 EPKTCNT 寄存器并保存其内容。
2. 读 ERXWRPTL 和 ERXWRPTH。
3. 再次读 EPKTCNT 寄存器。
4. 比较两个数据包计数。如果不相同，则返回步骤 2。

获得硬件写指针之后，剩余空间可以按例 18-2 中所示的方法计算。硬件禁止将写指针移动到 ERXRDPT 所指向的地址（除了配置缓冲区指针的情况），所以缓冲区中始终至少有一个字节未用。公式 18-2 中的计算反映了这一空出的字节。

公式 18-2： 接收缓冲区剩余空间计算

如果 ERXWRPT > ERXRDPT，则
    剩余空间 = (ERXND – ERXST) – (ERXWRPT – ERXRDPT)
否则
如果 ERXWRPT = ERXRDPT，则
    剩余空间 = (ERXND – ERXST)
否则
    剩余空间 = ERXRDPT – ERXWRPT – 1

表 18-5： 用于数据包发送的寄存器汇总

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	63
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	63
ESTAT	—	BUFER	—	LATECOL	—	RXBUSY	TXABRT	PHYRDY	63
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	60
ETXSTL	发送开始寄存器低字节 （ETXST<7:0>）								64
ETXSTH	—	—	—	发送开始寄存器高字节 （ETXST<12:8>）					64
ETXNDL	发送结束寄存器低字节 （ETXND<7:0>）								64
ETXNDH	—	—	—	发送结束寄存器高字节 （ETXND<12:8>）					64
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	65
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX	65
MACON4	—	DEFER	BPEN	NOBKOFF	—	—	r	r	65
MABBIPG	—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0	65
MAIPGL	—	非背对背包间间隔寄存器低字节 （MAIPGL<6:0>）							65
MAIPGH	—	非背对背包间间隔寄存器高字节 （MAIPGH<6:0>）							65
MACLCON1	—	—	—	—	重发最大数寄存器 （RETMAX<3:0>）				64
MACLCON2	—	—	冲突窗口寄存器 （COLWIN<5:0>）						64
MAMXFLL	最大帧长寄存器低字节 （MAMXFL<7:0>）								64
MAMXFLH	最大帧长寄存器高字节 （MAMXFL<15:8>）								64

图注： — = 未用， r = 保留位。未使用阴影单元。

表 18-6： 用于数据包接收的寄存器汇总

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	63
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	63
ESTAT	—	BUFER	—	LATECOL	—	RXBUSY	TXABRT	PHYRDY	63
ECON2	AUTOINC	PKTDEC	ETHEN	—	—	—	—	—	63
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	60
ERXSTL	接收开始寄存器低字节 （ERXST<7:0>）								64
ERXSTH	—	—	—	接收开始寄存器高字节 （ERXST<12:8>）					64
ERXNDL	接收结束寄存器低字节 （ERXND<7:0>）								64
ERXNDH	—	—	—	接收结束寄存器高字节 （ERXND<12:8>）					64
ERXRDPTL	接收缓冲区读指针低字节 （ERXRDPT<7:0>）								63
ERXRDPTH	—	—	—	接收缓冲区读指针高字节 （ERXRDPT<12:8>）					63
ERXFCON	UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN	64
EPKTCNT	以太网数据包计数寄存器								64
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSAL	MARXEN	65
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX	65
MACON4	—	DEFER	BPEN	NOBKOFF	—	—	r	r	65
MAMXFLL	最大帧长寄存器低字节 （MAMXFL<7:0>）								64
MAMXFLH	最大帧长寄存器高字节 （MAMXFL<15:8>）								64

图注： — = 未用， r = 保留位。未使用阴影单元。

18.6 双工模式配置与协商

以太网模块不支持自动双工模式协商。如果将模块连接到使能自动双工协商的网络交换机或以太网控制器上，模块将被检测为半双工设备。要在全双工模式下通信，必须手动将模块和远程节点（交换机、路由器或以太网控制器）配置为全双工操作方式。

18.6.1 半双工工作方式

当清零 FULDPX（MACON3<0>）和 PDPXMD（PHCON1<8>）位（= 0）时，以太网模块工作于半双工模式。如果仅置 1 这两个位的其中一位，模块将处于不确定的状态，无法正常工作。由于在全双工和半双工模式之间切换会造成这种不确定状态，所以在此期间应用程序不应发送任何数据包（保持 TXRTS 位清零），并禁止数据包接收（保持 RXEN 位清零）。

在半双工模式下，任何时候在物理介质上都只能有一个以太网控制器发送数据包。如果应用程序通过将 TXRTS 位置 1 来请求发送数据包，而此时已有另外一个以太网控制器正在发送，那么以太网模块将延迟，等待远程发送器停止。当远程发送器停止时，模块将尝试发送其数据包。如果正好有另外一个以太网控制器在近乎同一时间也启动发送，那么将造成线上的数据损坏，并发生冲突。

硬件将使用两种方法之一来处理这种情况。如果冲突发生时，发送字节数未达到 MACLCON2 中冲突窗口所指定的字节数，那么 TXRTS 位将保持置 1，并发生发送错误中断，然后在经过一段随机指数后退延时（由 IEEE 802.3 规范定义）之后，重新尝试从头开始发送数据包。应用程序不需要进行干预。如果重发尝试次数达到 MACLCON1 中定义的重发最大次数（RETMAX），那么将中止数据包发送，并清零 TXRTS 位。此时，应用程序就需要执行适当的操作。应用程序可以通过读 TXABRT 标志位确定数据包是被中止，而不是成功发送。更多信息，请参见**第 18.5.2 节 “发送数据包”**。

如果冲突发生时，发送字节数达到 MACLCON2 中冲突窗口所指定的字节数，那么将立即中止数据包，不再尝试任何重发操作。通常，如果正确配置了符合 802.3 规范的网络，将不会发生这种迟冲突。此时，可能需要用户干预来纠正该问题。该问题可能是由于全双工节点尝试在半双工介质上进行发送造成的。或者，模块可能在连接到全双工网络上时尝试工作于半双工模式。电缆过长或网络规模过大也可能会是造成迟冲突的原因。

当设置为半双工模式时，复位默认配置会将发送的数据包环回到发送节点。除非接收过滤器配置成将数据包过滤掉，否则它们会像其他网络通信数据一样被写入循环接收缓冲区中。为防止这种情况发生，需将 HDLDIS 位置 1。

18.6.2 全双工工作方式

当 FULDPX 和 PDPXMD 位都置 1（= 1）时，以太网模块工作于全双工模式。如果仅清零这两个位的其中一位，模块将处于不确定的状态，无法正常工作。重复前面所言，由于在全双工和半双工模式之间切换可能会造成这种不确定状态，在此期间应用程序不应发送任何数据包，并要禁止接收数据包。

在全双工模式下，可以在发送数据包的同时接收数据包。因此，发送数据包时不会造成任何冲突。所以将不使用一些配置字段，如 MACLCON1 中的重发最大数（RETMAX）字段和 MACLCON2 中的冲突窗口（COLWIN）字段。

当设置为全双工模式时，复位默认配置不会将发送的数据包环回到发送节点。如果需要利用环回来进行故障诊断，那么应用程序应将 PLOOPBK 位（PHCON1<14>）置 1。在全双工模式下使能环回时，将禁止双绞线输出驱动器，并忽略所有传入数据，从而将断开所有链路（如果已建立）。所有因为环回配置而接收到的数据包，都会像正常网络通信数据一样，需经过所有使能的接收过滤器过滤处理。

18.7 流量控制

以太网模块同时实现了全双工和半双工模式下的硬件流量控制。该功能的操作随所使用的模式而不同。

18.7.1 半双工模式

在半双工模式下，将 FCEN0 位（EFLOCON<0>）置 1 可以使能流量控制。当 FCEN0 置 1 时，将自动在以太网介质上持续不断地发送由交替的 1 和 0 组成的前导信号（55h）。所有连接节点都会检测到该数据，此时它们或者不发送任何数据、等待该发送结束，或者尝试发送数据并立即造成冲突。由于这种情况下始终会发生冲突，所以网络中的节点将无法互相通信，将不会收到任何新的数据包。

当应用程序通过将 TXRTS 位置 1 来使能模块发送数据包时，前导信号将停止发送。然后，在经过一个包间延时（由 MABBIPG 配置）后，模块开始尝试发送自己的数据包。在包间延时期间，其他节点可能会开始发送。由于前面的通信堵塞，所以可能会有几个节点同时开始发送，从而可能造成一系列的冲突。当模块成功完成数据包发送或中止了数据包发送时，前导信号会自动重新开始发送。当应用程序不希望再堵塞网络时，应该将 FCEN0 位清零。这样，前导信号将停止发送，恢复正常的网路操作。

考虑到半双工流量控制对网络的不利影响和低效率，除非是在经过正确测试的封闭网络环境下进行应用，否则建议不要使用半双工流量控制。

18.7.2 全双工模式

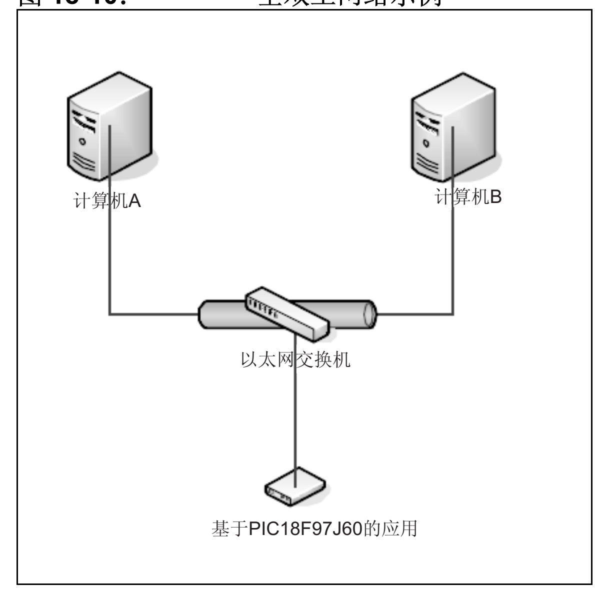
在全双工模式（MACON3<0> = 1）下，硬件流量控制通过发送暂停控制帧（由 IEEE 802.3 规范定义）来实现。暂停控制帧为 64 字节长，它包含保留的组播目标地址 01-80-C2-00-00-01、发送节点源地址、特殊的暂停操作码、2 字节的暂停定时器值和填充数据 /CRC。

通常，当 MAC 接收到暂停帧时，MAC 会完成正在发送的数据包，然后停止发送任何新的帧。暂停定时器值将被从控制帧中提取出来，用于对内部定时器进行初始化。每 512 比特时间（51.2 µs）定时器都会自动递减一次。当定时器在递减计数时，依然允许接收数据包。如果收到新的暂停帧，将用新的暂停定时器值重新初始化定时器。当定时器递减到 0 时，或者收到暂停定时器值为 0 的帧时，接收到暂停帧的 MAC 将恢复发送所有等待发送的数据包。为防止暂停帧造成整个网络上的所有通信中断，以太网交换机和路由器不会在全双工模式下传播暂停控制帧。暂停操作仅适用于接收节点。

图 18-10 给出了一个示例网络。在全双工模式下，如果计算机 A 要向基于单片机的应用发送太多的数据，以太网模块可以发送一个暂停控制帧来暂停向它发送的数据。以太网交换机将收到暂停帧并停止向单片机发送数据。如果计算机 A 继续发送数据，以太网交换机将对数据进行缓冲，以便稍后在暂停定时器结束时再进行发送。如果以太网交换机缓冲区空间出现不足，它通常会向计算机 A 发送一个它自己的暂停控制帧。

如果由于某种原因，以太网交换机没有生成自己的暂停控制帧，或者某个节点未正确处理所接收到的暂停帧，那么数据包将不可避免地丢失。在任何情况下，计算机 A 和计算机 B 之间的通信完全不会受到影响。

图 18-10： 全双工网络示例



要使能全双工模式下的流量控制，需要将 MACON1 寄存器中的 EXPAUS 和 RXPAUS 位置 1。然后，每当接收缓冲区空间用尽时，置 1 流量控制使能位 FCEN1:FCEN0（EFLOCON<1:0>）。模块将会自动完成正在发送的所有数据，然后发送一个载有合适定时器值的有效暂停帧。根据所选择的模式，应用程序最终可能需要通过再次写 FCEN 位来清除流量控制模式。

当 RXPAUS 置 1，并且收到具有非 0 暂停定时器值的有效暂停帧时，模块将自动禁止发送。如果 TXRTS 位被置 1 而需要发送一个数据包，硬件将一直等到暂停定时器定时结束，然后再尝试发送该数据包，并随后清零 EXRTS 位。通常，这些对于单片机来说是透明的，单片机永远不会知道接收到了一个暂停帧。如果希望知道 MAC 是否暂停，用户应该将 PASSALL 位（MACON1<1>）置 1，然后手动分析可能会收到的暂停控制帧。

寄存器 18-20： EFLOCON：以太网流量控制寄存器

U-0	U-0	U-0	U-0	U-0	R-0	R/W-0	R/W-0
—	—	—	—	—	FULDPXS	FCEN1	FCEN0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-3　未用：读为 0

bit 2　**FULDPXS：**只读 MAC 全双工影子位
1 = MAC 配置为全双工模式，FULDPX（MACON3<0>）置 1
0 = MAC 配置为半双工模式，FULDPX（MACON3<0>）清零

bit 1-0　**FCEN1:FCEN0：**流量控制使能位
当 FULDPXS = 1 时：
11 = 发送一个定时器值为 0 的暂停帧，然后关闭流量控制
10 = 周期性地发送暂停帧
01 = 发送一个暂停帧，然后关闭流量控制
00 = 流量控制关闭
当 FULDPXS = 0 时：
11 = 流量控制打开
10 = 流量控制关闭
01 = 流量控制打开
00 = 流量控制关闭

表 18-7： 用于流量控制的寄存器汇总

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	60
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	65
MABBIPG	—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0	65
EFLOCON	—	—	—	—	—	FULDPXS	FCEN1	FCEN0	65
EPAUSL	暂停定时器值寄存器低字节（EPAUS<7:0>）								65
EPAUSH	暂停定时器值寄存器高字节（EPAUS<15:8>）								65

图注：　— = 未用，r = 保留位。未使用阴影单元。

18.8 接收过滤器

为了最大程度地减小单片机处理工作量，以太网模块在内部实现了一系列不同的接收过滤器，可以自动拒绝不需要的数据包。总共实现了 6 种不同类型的数据包过滤器：

- 单播
- 组播
- 广播
- 格式匹配
- Magic Packet™
- 哈希表

各个过滤器均通过ERXFCON 寄存器 （寄存器 18-21）进行配置。任何时间均可以有一个以上的过滤器同时有效。此外，还可以通过 ANDOR 位对几个过滤器的测试结果进行逻辑 “与”或者逻辑 “或”运算。也就是说，可以设置为只有当所有有效的过滤器都接受才接受数据包，或者只要有一个过滤器接受就接受数据包。图 18-11 和图 18-12 中的流程图显示了不同 ANDOR 设置情况下每个过滤器的作用。

器件可以通过清零 ERXFCON 寄存器来进入混杂模式（Promiscuous mode），并接收所有数据包。用户需要根据应用的要求对寄存器进行适当的设置。

18.8.1 单播过滤器

单播接收过滤器会检查所有传入数据包的目标地址。如果目标地址与 MAADR 寄存器中的内容完全匹配，数据包就满足单播过滤器的条件。

18.8.2 组播过滤器

组播接收过滤器会检查所有传入数据包的目标地址。如果目标地址首字节的最低有效位置 1，数据包就满足组播过滤器的条件。

18.8.3 广播过滤器

广播接收过滤器会检查所有传入数据包的目标地址。如果目标地址为 FF-FF-FF-FF-FF-FF，那么数据包满足广播过滤器的条件。

18.8.4 哈希表过滤器

哈希表接收过滤器会根据数据包中的 6 个目标地址字节计算 CRC。然后 CRC 用作指针指向 EHT 寄存器中的位。如果指针所指位置 1，那么数据包满足哈希表过滤器的条件。例如，如果计算得到的 CRC 为 05h，则检查哈希表中的 bit 5。如果该位置 1，则满足哈希表过滤器的条件。如果哈希表中的每个位都是零，数据包将永远不会满足过滤器的条件。类似地，如果哈希表中的每个位都置 1，数据包将总是满足过滤器的条件。

寄存器 18-21： ERXFCON：以太网接收过滤器控制寄存器

R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **UCEN**：单播过滤器使能位
当 ANDOR = 1 时：
1 = 目标地址与本地 MAC 地址不匹配的数据包将被丢弃
0 = 禁止过滤器
当 ANDOR = 0 时：
1 = 接受目标地址与本地 MAC 地址匹配的数据包
0 = 禁止过滤器

bit 6 **ANDOR**：与 / 或过滤器选择位
1 = 与：只有所有使能的过滤器都接受时才接受数据包，否则拒绝数据包
0 = 或：只要有任一使能的过滤器接受时就接受数据包，只有全部拒绝时才拒绝

bit 5 **CRCEN**：后过滤器 CRC 校验使能位
1 = 丢弃所有具有无效 CRC 的数据包
0 = 忽略 CRC

bit 4 **PMEN**：格式匹配过滤器使能位
当 ANDOR = 1 时：
1 = 数据包必须满足格式匹配条件，否则将被丢弃
0 = 禁止过滤器
当 ANDOR = 0 时：
1 = 接受满足格式匹配条件的数据包
0 = 禁止过滤器

bit 3 **MPEN**：Magic Packet 过滤器使能位
当 ANDOR = 1 时：
1 = 数据包必须为本地 MAC 地址的 Magic Packet，否则将被丢弃
0 = 禁止过滤器
当 ANDOR = 0 时：
1 = 接受本地 MAC 地址的 Magic Packet
0 = 禁止过滤器

bit 2 **HTEN**：哈希表过滤器使能位
当 ANDOR = 1 时：
1 = 数据包必须满足哈希表条件，否则将被丢弃
0 = 禁止过滤器
当 ANDOR = 0 时：
1 = 接受满足哈希表条件的数据包
0 = 禁止过滤器

bit 1 **MCEN**：组播过滤器使能位
当 ANDOR = 1 时：
1 = 数据包中目标地址的最低有效位必须置 1，否则数据包将被丢弃
0 = 禁止过滤器
当 ANDOR = 0 时：
1 = 接受目标地址最低有效位置 1 的数据包
0 = 禁止过滤器

bit 0 **BCEN**：广播过滤器使能位
当 ANDOR = 1 时：
1 = 数据包的目标地址必须为 FF-FF-FF-FF-FF-FF，否则将被丢弃
0 = 禁止过滤器
当 ANDOR = 0 时：
1 = 接受目标地址为 FF-FF-FF-FF-FF-FF 的数据包
0 = 禁止过滤器

图 **18-11**： 采用或逻辑的接收过滤器

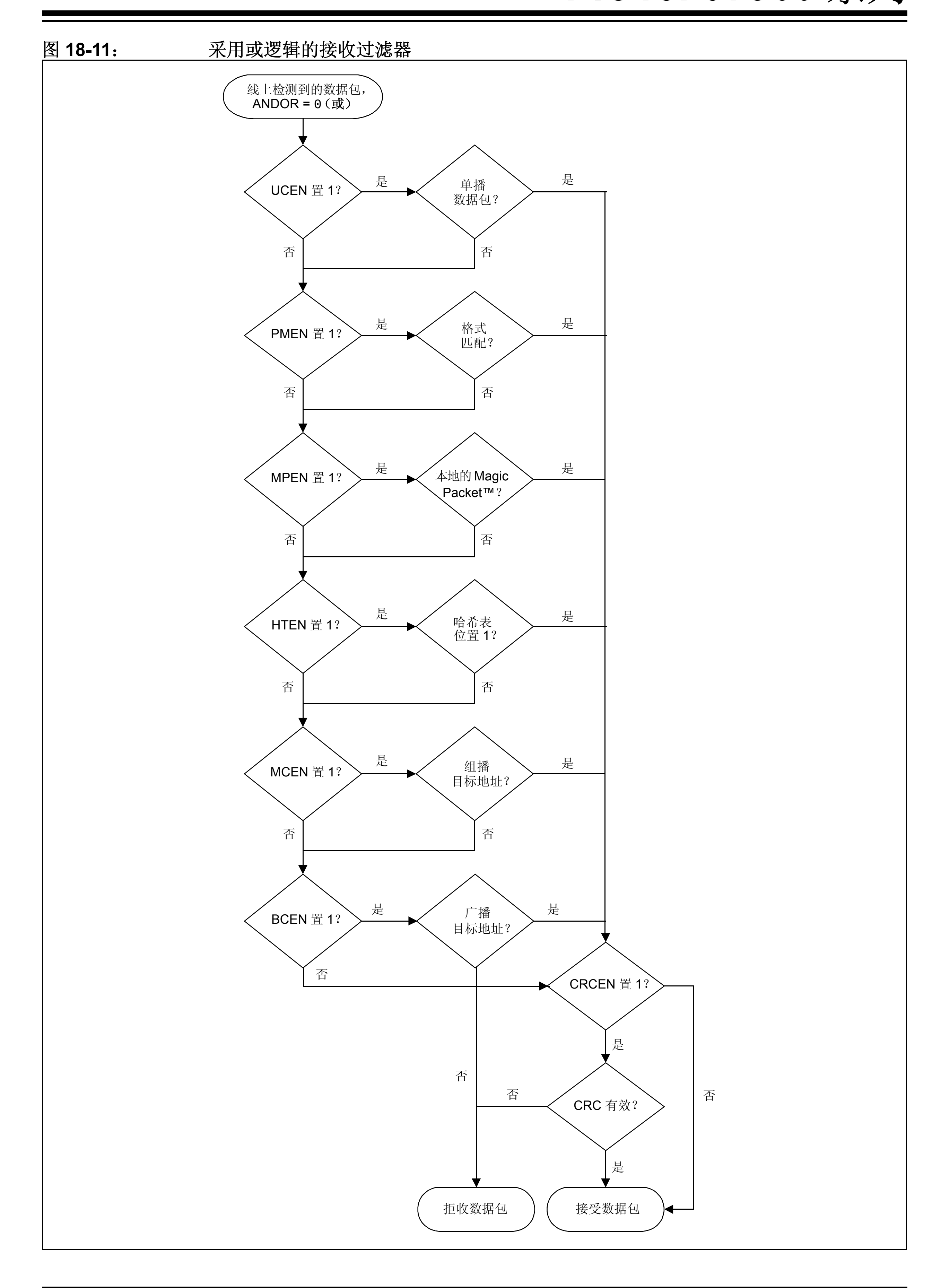
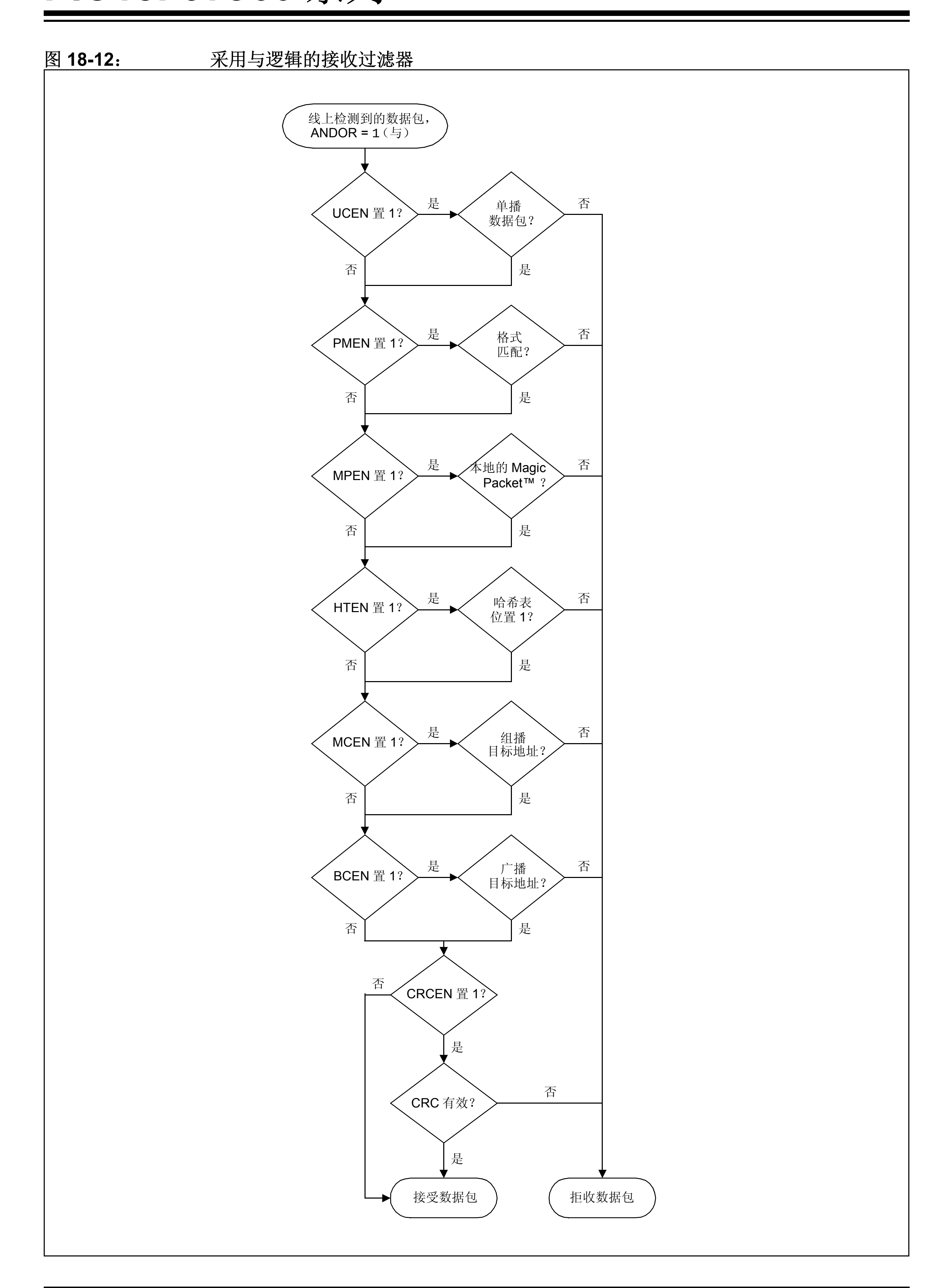


图 18-12： 采用与逻辑的接收过滤器



 超前信息

18.8.5 格式匹配过滤器

格式匹配过滤器会从传入数据包中选择 64 个字节，并计算字节的 IP 校验和，然后将校验和与 EPMCS 寄存器进行比较。如果计算的校验和与 EPMCS 寄存器匹配，则数据包满足格式匹配过滤器的条件。格式匹配过滤器通常用于过滤内部包含期望数据的数据包。

要使用格式匹配过滤器，应用程序必须设定格式匹配偏移量（EPMOH:EPMOL）、所有格式匹配屏蔽字节（EPMM0:EPMM7）和格式匹配校验和寄存器对（EPMCSH:EPMCSL）。格式匹配偏移量应等于从目标地址字段起始处到 64 字节窗口（用于计算校验和）的偏移。对于处于 64 字节窗口内的各个字节，通过在格式匹配屏蔽字节中置 1 或清零它们的相应位，可以有选择地将其包括在校验和计算内或排除在校验和计算外。如果接收到的数据包导致 64 字节窗口超出 CRC 的末尾，将立即导致不满足过滤器的条件，即便是相应的屏蔽位都为 0。格式匹配校验和寄存器应该设定为选定字节的校验和。校验和的计算方法与 DMA 模块计算校验和的方法相同（见**第 18.9.2 节“校验和计算”**）。相应屏蔽位设定为 0 的数据字节将从校验和计算中完全排除，而不是将数据字节视为 0。

例如，如果应用程序要过滤所有具有特定源 MAC 地址 00-04-A3-FF-FF-FF 的数据包，可以将格式匹配偏移量设定为 0000h，然后将 EPMM0 的 bit 6、bit 7，以及 EPMM1 的 bit 0、bit 1、bit2 和 bit 3 置 1（假设所有其他的屏蔽位均为 0）。EPMCS 寄存器中的校验和应编程为 5BFCh。还有一种配置方法，可以将偏移量设定为 0006h，并将 EPMM0 的 bit 0、bit 1、bit 2、bit 3、bit 4 和 bit 5 置 1。校验和将仍为 5BFCh。不过，第二种方法有些不太理想，因为当数据包长度小于 70 字节时，虽然它们同样可以生成正确的校验和，但却永远不会满足格式匹配的条件。

图 18-13 给出了格式匹配过滤器的另外一个示例。

图 18-13： 格式匹配示例

输入配置：

EMPOH:EPMOL = 0006h
EPMM0:EPMM7 = 0000000000001F0Ah
EPMCSH:EPMCSL = 563Fh

字段	DA	SA	类型 / 长度	数据	FCS
接收的数据	11 22 33 44 55 66	77 88 99 AA BB CC	00 5A	09 0A 0B 0C 0D . . . 40 . . .	FE 45 23 01
字节编号	0 1 2 3 4 5	6 7 8 9 10 11	12 13	14 15 16 17 18 . . . 70 . . .	

用于计算校验和的字节：7、9、14、15、16、17、18

用于格式匹配的 64 字节窗口

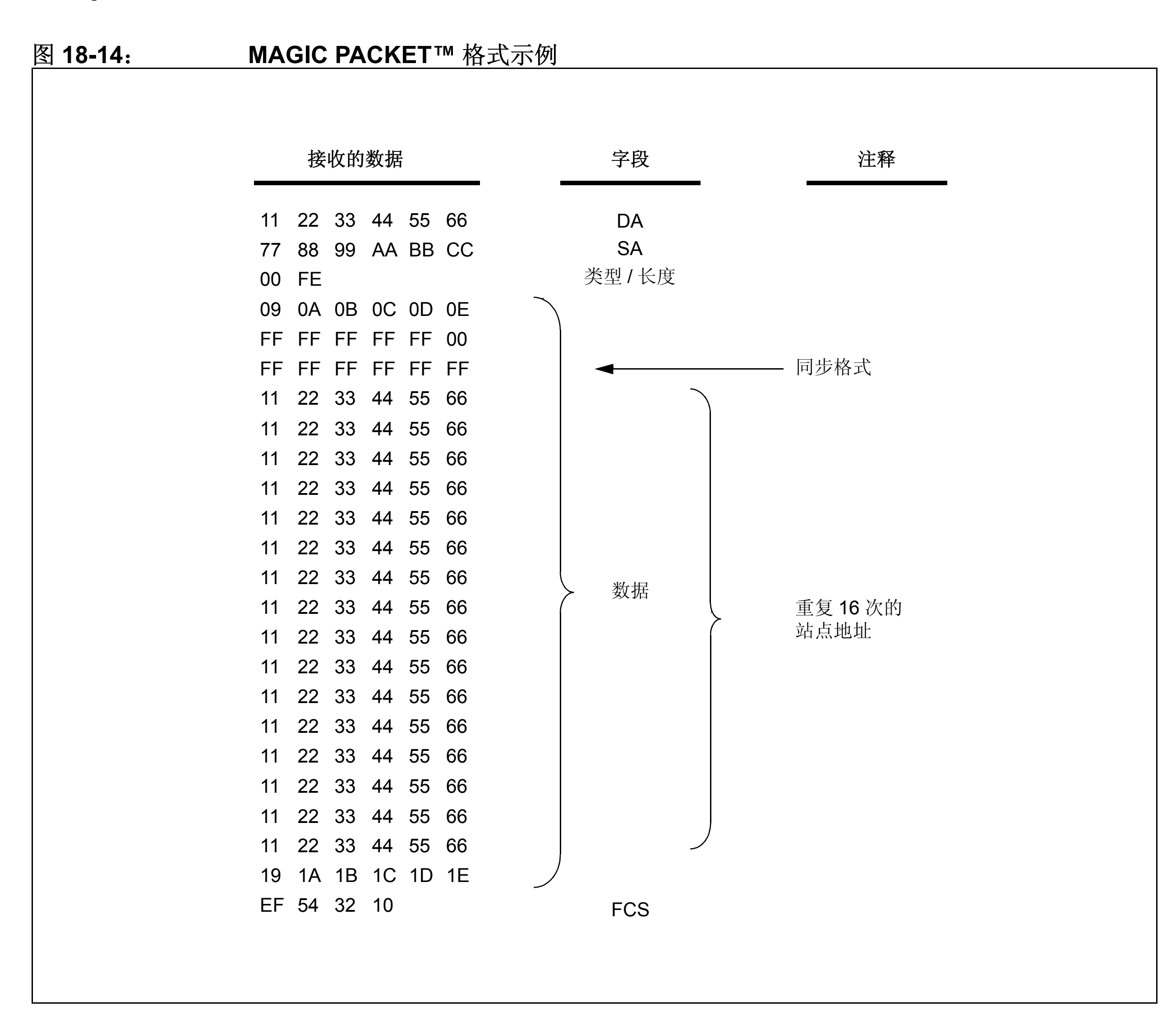
用于计算校验和的值 = {88h, AAh, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 00h}
（00h 是由硬件添加的填充字节）

注： 接收数据以十六进制显示。字节编号以十进制格式显示。

18.8.6 MAGIC PACKET 过滤器

Magic Packet 格式由一个包含 6 个 FFh 字节的同步格式和重复 16 次的目标地址 （图 18-14）组成。 Magic Packet 过滤器会检查所有传入数据包的目标地址和数据字段。如果目标地址与 MAADR 寄存器匹配，并且数据字段内包含有效的Magic Packet格式，那么数据包满足 Magic Packet 过滤器的条件。

图 18-14： MAGIC PACKET™ 格式示例



超前信息

18.9 直接存储器访问控制器

以太网模块内部包含一个双功能的DMA控制器，它可以用来在8 KB缓冲存储器中复制数据；也可以用来计算兼容各种工业标准通信协议（包括 TCP、UDP、IP 和 ICMP 等）的 16 位校验和。

可通过 3 个指针和一些状态 / 控制位来控制 DMA：

- EDMASTH:EDMASTL——源起始地址
- EDMANDH:EDMANDL——源结束地址
- EDMADSTH:EDMADSTL——目标起始地址
- ECON1<DMAST> 和 ECON1<CSUMEN>
- EIR<DMAIF> 和 EIE<DMAIE>

源指针和结束指针定义要进行复制或计算校验和的数据。目标指针仅在复制数据时使用，它定义被复制数据的存放位置。3 个指针都只用于 8 KB 以太网存储器，不能用来访问 PIC® 单片机数据存储器空间。

当 DMA 操作开始时，EDMAST 寄存器对被复制到内部源指针中。DMA 将一次执行一个字节，然后递增内部源指针。不过，如果在处理一个字节之后，内部源指针与接收缓冲区结束指针 ERXND 相等，那么源指针不会递增。此时，内部源指针将装入接收缓冲区起始指针 ERXST 的值。这样， DMA 将按照接收缓冲区的循环 FIFO 结构进行操作，从而接收到的数据包可以遵循同一种方式进行处理。当内部源指针与 EDMAND 指针匹配时， DMA 操作将结束。

当正在进行任何 DMA 操作时，不应修改 DMA 指针和 CSUMEN 位（ECON1<4>）。任何时候都可以通过清零 DMAST 位（ECON1<5>）来取消 DMA 操作。所有寄存器的值都不会改变；不过，如果取消前正在进行 DMA 复制，则可能有一些存储器字节已经被复制。使用 DMA 时，应注意一些实施注意事项和固件要求 （见注 1）。

> 注 1：如果由于接收缓冲区折回操作的原因而无法到达 EDMAND 指针处，DMA 操作将永远不会结束。
>
> 2：根据设计要求，DMA 模块不能用来仅对一个字节（EDMAST = EDMAND）进行复制或校验和计算。如果尝试这样操作，则可能会覆盖缓冲区中的所有存储单元，并且永远不会停止。
>
> 3：DMA 操作结束（DMAST 由硬件或软件清零）之后，固件绝对不能在 4 个指令周期内将 DMAST 位再次置 1。

18.9.1 复制存储器内容

要复制缓冲区中的存储器内容：

1. 使用合适的起始、结束和目标地址设定 EDMAST、 EDMAND 和 EDMADST 寄存器对。EDMAST 寄存器应指向要复制数据的第一个字节，EDMAND 寄存器应指向要复制数据的最后一个字节，而 EDMADST 寄存器应指向目标地址范围内的第一个字节。目标地址范围始终为线性的，除了从 8191 返回到 0 （8 KB 存储器边界）之外，不会在任何地方发生折回。在计算结束指针时，要特别小心，以防止出现会覆盖整个 8 KB 缓冲区的永远无法结束的 DMA 操作。
2. 如果需要，可以将 DMAIE （EIE<5>）和 ETHIE （PIE2<5>）位置 1，并清零 DMAIF （EIR<5>）标志位，以允许在复制过程结束时产生中断。
3. 清零 CSUMEN （ECON1<4>）位。
4. 通过将 DMAST （ECON1<5>）位置 1 来启动 DMA 复制操作。

如果将 DMAST 位置 1 时正在进行发送操作 （TXRTS 位置 1），那么模块将等待至发送操作完成之后，再尝试进行 DMA 复制。这个延时是必要的，因为 DMA 和发送引擎不能同时访问缓冲区。

完成复制之后，DMA 硬件将清零 DMAST 位，将 DMAIF 位置 1，并产生中断（如果允许）。指针和 EDMACS 寄存器将不变。

在 DMA 模块已经初始化并开始复制操作之后，复制每个字节都需要 1 个指令周期 （TCY）。不过，如果 CPU 内核对 EDATA 寄存器执行读或写指令，或者以太网接收硬件累积了一个字节的数据，DMA 将停止该周期，让步于高优先级的操作。如果复制一个 1518 字节 （最大长度）的数据包，同时没有任何其他存储器使用带宽，那么在内核频率 41.667 MHz 下，DMA 模块将需要略长于 145.7 µs 的时间来完成复制。复制一个最小大小（64 字节）的数据包大约需要 6.2 µs 的时间 （41.667 MHz 下）加上寄存器配置时间。

18.9.2 校验和计算

校验和计算逻辑将源数据视为一系列16位大尾整数。如果源数据个数为奇数个字节，为了计算校验和，将会在数据末尾追加一个填充字节 （00h）。

计算得到的校验和是所有 16 位整数补码和的补码。例如，如果校验和中包含的字节为 {89h, ABh, CDh}，那么计算校验和时首先计算 89ABh + CD00h。在示例中，第 16 位会发生进位，所以在计算 16 位反码时，进位将加到第 1 位上。对得到的结果值 56ACh 求反码，最终将得到校验和 A953h。

要计算校验和：

1. 设置 EDMAST 和 EDMAND 寄存器对，使它们分别指向校验和中要包括的缓冲区数据的第一和最后一个字节。在设定这些指针时，要特别小心，避免出现由于接收缓冲区折回而引起的永远无法结束的校验和计算。
2. 如果要在计算完成时产生中断，可以将 DMAIE （EIE<5>）和 ETHIE （PIE2<5>）位置 1，并清零 DMAIF （EIR<5>）位。
3. 通过将 CSUMEN （ECON1<4>）和 DMAST （ECON1<5>）位置 1 来启动计算。

完成校验和计算之后，硬件将清零 DMAST 位，将 DMAIF 位置 1，并产生中断 （如果允许）。 DMA 指针将保持不变，不会将数据写入任何存储单元。计算得到的校验和将保存在EDMACSH与EDMACSL寄存器中。应用程序可以将该值写入一个数据包、将其与 0 进行比较 （以验证接收到的数据块中是否包含校验和字段），或将其与其他一些校验和进行比较，例如各种协议 （TCP、 UDP 等）中所使用的伪报头校验和。

各种协议 （例如 TCP 和 IP）都会在校验和所涵盖的数据范围内包含一个校验和字段。如果接收到需要进行验证的数据包，应用程序可以执行以下操作：

1. 从数据包中读校验和，并将其保存在一个临时单元。
2. 向校验和字段写入 0。
3. 使用 DMA 控制器计算新的校验和。
4. 将结果与步骤 1 保存的校验和进行比较。

当通过 ERXRDPT 指针保护写地址之后，写接收缓冲区是允许的。更多信息，请参见**第 18.5.3 节 “接收数据包”**。

IP 校验和具有唯一的数学特性，可以用来在某些情况下进一步减少处理工作量。在某些应用中，可能并不需要将数据写入接收缓冲区。

当 DMA 工作于校验和模式下时，处理校验和中包含的每个字节都需要一个指令周期 （TCY）。因而，如果对超过 1446 字节的数据执行校验和计算，在 41.667 MHz 下DMA模块将需要略长于138.8 µs的时间来完成操作。

在相同频率下，一个较小的 20 字节的报头字段需要的时间大约是 1.9 µs 加上 DMA 设置时间。这些估计时间是在这样的假设下得出的：假设以太网接收硬件不需要存储器访问带宽，并且当 DMA 在进行计算时 CPU 内核不会向 EDATA 寄存器发出任何读或写命令。

与 DMA 复制模式一样，只有 TXRTS 位（ECON1<3>）清零时，校验和操作才会开始。如果应用程序刚刚发送很大的数据包，并立即对一个接收到的数据包进行校验和验证，这将会在一定程度上增加校验和计算时间。

表 18-8： 与 DMA 控制器相关的寄存器汇总

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	63
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	63
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	60
ERXNDL	接收结束寄存器低字节 （ERXND<7:0>）								63
ERXNDH	—	—	—	接收结束寄存器高字节 （ERXND<12:8>）					63
EDMASTL	DMA 开始寄存器低字节 （EDMAST<7:0>）								63
EDMASTH	—	—	—	DMA 开始寄存器高字节 （EDMAST<12:8>）					63
EDMANDL	DMA 结束寄存器低字节 （EDMAND<7:0>）								63
EDMANDH	—	—	—	DMA 结束寄存器高字节 （EDMAND<12:8>）					63
EDMADSTL	DMA 目标寄存器低字节 （EDMADST<7:0>）								63
EDMADSTH	—	—	—	DMA 目标寄存器高字节 （EDMADST<12:8>）					63
EDMACSL	DMA 校验和寄存器低字节 （EDMACS<7:0>）								63
EDMACSH	DMA 校验和寄存器高字节 （EDMACS<15:8>）								63

图注： — = 未用。未使用阴影单元。

18.10 模块复位

以太网模块提供了以下模块复位方式：

- 仅发送复位
- 仅接收复位

18.10.1 上电复位 （POR）

当 VDD 的电平达到工作要求时，以太网模块同时使用单片机的上电复位脉冲来进入初始化状态。器件规定了 VDD 的最小上升速率 （见**第 27.0 节 "电气特性"**）。

上电复位之后，以太网缓冲存储器中的内容处于未知状态。所有 SFR 和 PHY 寄存器都将装入其规定的复位值。不过，只有在 PHY 起振定时器定时结束并且 PHYRDY 位 （ESTAT<0>）置 1，或者 ETHEN 位置 1 超过 1 ms 时，才可以访问 PHY 寄存器。更多详细信息，请参见**第 18.1.3.1 节 "起振定时器"**。

18.10.2 仅发送复位

仅发送复位通过将 TXRST 位（ECON1<7>）置 1 来实现。它仅对发送逻辑进行复位。其他寄存器和控制电路（例如缓冲区管理和主机接口）不会受仅发送复位影响。要恢复为正常操作，可以用软件清零 TXRST 位。

18.10.3 仅接收复位

仅接收复位通过将 RXRST 位 （ECON1<6>）置 1 来实现。该操作仅对接收逻辑进行复位。其他寄存器和控制电路 （例如缓冲区管理和主机接口）不会受仅接收复位影响。要恢复为正常操作，可以用软件清零 RXRST 位。

注：

19.0 主控同步串行口 （MSSP）模块

19.1 主控 SSP （MSSP）模块概述

主控同步串行口 （MSSP）模块是用于同其他外设或单片机器件进行通信的串行接口。这些外设器件可以是串行 EEPROM、移位寄存器、显示驱动器和 A/D 转换器等。 MSSP 模块有下列两种工作模式：

- 串行外设接口 （Serial Peripheral Interface，SPI）
- I²C™
  - 全主控模式
  - 从动模式 （支持广播地址呼叫）

I²C 接口硬件上支持下列模式：

- 主控模式
- 多主机模式
- 从动模式

PIC18F97J60 系列的 64 引脚和 80 引脚器件有一个 MSSP 模块，称为 MSSP1。100 引脚器件有两个 MSSP 模块，称为 MSSP1 和 MSSP2。每个模块都独立工作。

注：	在本节中，在所有工作模式下，通常指的 MSSP 模块都可以解释为 MSSP1 或 MSSP2。寄存器名称和模块 I/O 信号使用通用标识符 “x”（数字）来区分某个特定模块。控制位名称没有区别。

19.2 控制寄存器

每个 MSSP 模块有三个相关的控制寄存器，包括一个状态寄存器（SSPxSTAT）和两个控制寄存器（SSPxCON1 和 SSPxCON2）。根据 MSSP 模块是在 SPI 模式还是 I²C 模式下工作，这些寄存器的用途及它们各自的配置位将完全不同。

下面各节会提供更多详细信息。

注：	在具有多个 MSSP 模块的器件中，要特别注意 SSPxCON 寄存器名称。SSP1CON1 和SSP1CON2控制同一模块工作的不同方面，而 SSP1CON1 和 SSP2CON1 控制两个不同模块的相同功能。

19.3 SPI 模式

SPI 模式允许同时同步发送和接收 8 位数据。器件支持 SPI 的所有四种模式。通常使用以下三个引脚来实现通信：

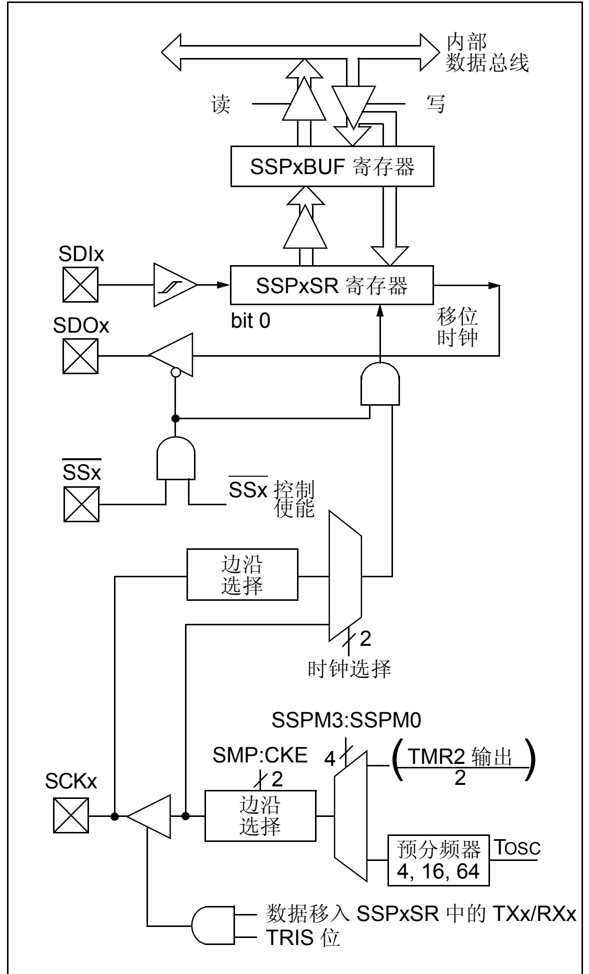
- 串行数据输出 （Serial Data Out， SDOx）—— RC5/SDO1 （或 RD4/SDO2，适用于 100 引脚器件）
- 串行数据输入 （Serial Data In， SDIx）—— RC4/SDI1/SDA1 （或 RD5/SDI2/SDA2，适用于 100 引脚器件）
- 串行时钟 （Serial Clock， SCKx）—— RC3/SCK1/SCL1 （或 RD6/SCK2/SCL2，适用于 100 引脚器件）

此外，当处于从动工作模式时要使用第 4 根引脚：

- 从动选择 （Slave Select， $\overline{\text{SSx}}$）——RF7/$\overline{\text{SS1}}$ （或 RD7/$\overline{\text{SS2}}$，适用于 100 引脚器件）

图 19-1 给出了 MSSP 模块在 SPI 模式下的工作原理框图。

图 19-1： MSSP 框图 （SPI 模式）



19.3.1 寄存器

MSSP 模块有四个寄存器用于 SPI 工作模式。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPxCON1）
- MSSP 状态寄存器 （SSPxSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPxBUF）
- MSSP 移位寄存器（SSPxSR）——不可直接访问

SSPxCON1 和 SSPxSTAT 是 SPI 模式下的控制寄存器和状态寄存器。SSPxCON1寄存器是可读写的。SSPxSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPxSR 是用来将数据移入或移出的移位寄存器。SSPxBUF 是缓冲寄存器，可用于数据字节的写入或读出。

接收数据时，SSPxSR 和 SSPxBUF 共同构成一个双重缓冲接收器。当 SSPxSR 接收到一个完整的字节之后，该字节会被送入 SSPxBUF，同时将中断标志位 SSPxIF 置 1。

在数据发送过程中， SSPxBUF 不是双重缓冲的，对 SSPxBUF的写操作将同时写入SSPxBUF和SSPxSR。

寄存器 19-1： SSPxSTAT：MSSPx 状态寄存器 （SPI 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE(1)	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SMP**：采样位
SPI 主控模式：
1 = 在数据输出时间的末端采样输入数据
0 = 在数据输出时间的中间采样输入数据
SPI 从动模式：
当 SPI 工作在从动模式时，必须将 SMP 清零。

bit 6 **CKE**：SPI 时钟选择位 (1)
1 = 时钟状态从有效转换到空闲时发送
0 = 时钟状态从空闲转换到有效时发送

bit 5 **D/$\overline{A}$**：数据 / 地址位
只在 I^2C 模式下使用。

bit 4 **P**：停止位
只在 I^2C 模式下使用。当禁止 MSSP 模块 （SSPEN 清零）时，该位被清零。

bit 3 **S**：启动位
只在 I^2C 模式下使用。

bit 2 **R/$\overline{W}$**：读 / 写信息位
只在 I^2C 模式下使用。

bit 1 **UA**：更新地址位
只在 I^2C 模式下使用。

bit 0 **BF**：缓冲区满状态位 （仅用于接收模式）
1 = 接收完成， SSPxBUF 满
0 = 接收未完成， SSPxBUF 空

注 1： 时钟状态的极性由 CKP 位 （SSPxCON1<4>）设置。

寄存器 19-2： SSPxCON1：MSSPx 控制寄存器 1 （SPI 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(3)	SSPM2(3)	SSPM1(3)	SSPM0(3)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **WCOL**：写冲突检测位 （仅用于发送模式）
1 = 正在发送前一个字时，又有数据写入 SSPxBUF 寄存器 （必须用软件清零）
0 = 未发生冲突

bit 6 **SSPOV**：接收溢出指示位 (1)
SPI 从动模式：
1 = SSPxBUF 中仍保存前一数据时，又接收到一个新的字节。如果发生溢出，SSPxSR 中的数据会丢失。溢出只会在从动模式下发生。即使只是发送数据，用户也必须读 SSPxBUF，以避免将溢出标志位置 1 （该位必须用软件清零）。
0 = 无溢出

bit 5 **SSPEN**：主控同步串口使能位 (2)
1 = 使能串口并将 SCKx、 SDOx、 SDIx 和 $\overline{\text{SSx}}$ 配置为串口引脚
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚

bit 4 **CKP**：时钟极性选择位
1 = 空闲状态时，时钟为高电平
0 = 空闲状态时，时钟为低电平

bit 3-0 **SSPM3:SSPM0**：主控同步串口模式选择位 (3)
0101 = SPI 从动模式，时钟 = SCKx 引脚，禁止 $\overline{\text{SSx}}$ 引脚控制，可将 $\overline{\text{SSx}}$ 用作 I/O 引脚
0100 = SPI 从动模式，时钟 = SCKx 引脚，使能 $\overline{\text{SSx}}$ 引脚控制
0011 = SPI 主控模式，时钟 = TMR2 输出 /2
0010 = SPI 主控模式，时钟 = Fosc/64
0001 = SPI 主控模式，时钟 = Fosc/16
0000 = SPI 主控模式，时钟 = Fosc/4

注 1： 在主控模式下，溢出位不会被置 1，因为每次接收 （和发送）新数据都是通过写入 SSPxBUF 寄存器启动的。
2： 当使能时，必须将这些引脚正确地配置为输入或输出。
3： 在此未列出的位组合用于保留或仅在 I^2C™ 模式下使用。

19.3.2 工作原理

当初始化 SPI 时，需要指定几个选项。可以通过编程相应的控制位 （SSPxCON1<5:0> 和 SSPxSTAT<7:6>）来指定。这些控制位用于指定以下选项：

- 主控模式 （SCKx 作为时钟输出）
- 从动模式 （SCKx 作为时钟输入）
- 时钟极性 （SCKx 的空闲状态）
- 数据输入采样相位 （数据输出时间的中间或末尾）
- 时钟边沿 （在 SCKx 的上升沿 / 下降沿输出数据）
- 时钟速率 （仅用于主控模式）
- 从动选择模式 （仅用于从动模式）

MSSP 模块由一个发送 / 接收移位寄存器 （SSPxSR）和一个缓冲寄存器 （SSPxBUF）组成。 SSPxSR 将数据移入 / 移出器件，最高有效位在前。在新数据接收完毕前，SSPxBUF 保存上次写入 SSPxSR 的数据。一旦 8 位数据接收完毕，该字节就被移入 SSPxBUF 寄存器。然后，缓冲区满检测位 BF（SSPxSTAT<0>）和中断标志位 SSPxIF 被置 1。这种双重缓冲数据接收方式（SSPxBUF），允许在 CPU 读取刚接收的数据之前，就开始接收下一个字节。在数据发送 / 接收期间，任何试图写 SSPxBUF 寄存器的操作都无效，并且写冲突检测位 WCOL（SSPxCON1<7>）将被置 1。用户必须用软件将 WCOL 位清零才能判断以后对 SSPxBUF 寄存器的写入是否成功。

为确保应用软件能有效地接收数据，在下一个要发送的数据字节写入 SSPxBUF 之前，读取 SSPxBUF 中现有的数据。缓冲区满位 BF （SSPxSTAT<0>）用于表示何时 SSPxBUF 装入了接收到的数据 （发送完成） 。当 SSPxBUF 中的数据被读取后，BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会该数据。通常，可用 MSSP 中断来判断发送 / 接收是否已完成。必须读取和 / 或写入 SSPxBUF。如果不打算使用中断，用软件查询的方法同样可确保不会发生写冲突。例 19-1 举例说明了装载 SSP1BUF （SSP1SR）进行数据发送的过程。

不能直接读写SSPxSR寄存器，只能通过寻址SSPxBUF寄存器来访问。此外，SSPxSTAT 寄存器用于指示各种状态。

例 19-1： 装载 SSP1BUF （SSP1SR）寄存器

```
LOOP    BTFSS   SSP1STAT, BF    ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSP1BUF, W      ;WREG reg = contents of SSP1BUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSP1BUF         ;New data to xmit
```


19.3.3 使能 SPI I/O

要使能串口，MSSP 使能位 SSPEN（SSPxCON1<5>）必须置 1。要复位或重新配置 SPI 模式，要先将 SSPEN 位清零，重新初始化 SSPxCON 寄存器，然后将 SSPEN 位置 1。这将把 SDIx、SDOx、SCKx 和 SSx 引脚配置为串口引脚。要让上述引脚充当串口，必须正确设置引脚的数据方向位 （在 TRIS 寄存器中）。

- SDIx 由 SPI 模块自动控制
- SDOx 必须将 TRISC<5> （或 TRISD<4>）位清零
- SCKx （主控模式）必须将 TRISC<3> （或 TRISD<6>）位清零
- SCKx （从动模式）必须将 TRISC<3> （或 TRISD<6>）位置 1
- $\overline{\text{SS}}$x 必须将 TRISF<7> （或 TRISD<7>）位置 1

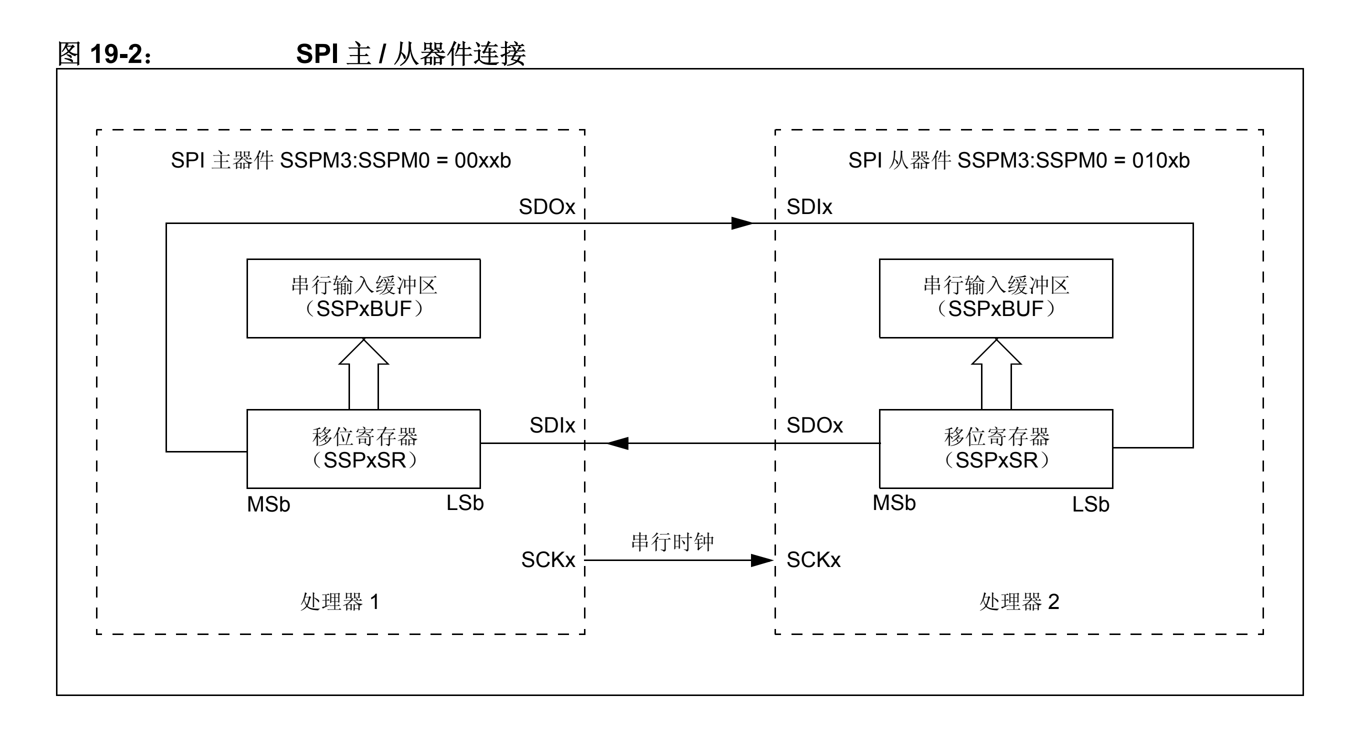
对于不需要的串口功能，可通过将对应的数据方向寄存器 （TRIS）设置为相反值来屏蔽。

19.3.4 典型连接

图 19-2 给出了两个单片机之间的典型连接。主器件（处理器 1）通过发送 SCKx 信号来启动数据传输。在两个处理器的移位寄存器之间，数据在编程设定的时钟边沿被传送，并在相反的时钟边沿被锁存。必须将两个处理器的时钟极性（CKP）设置为相同，这样就可以同时收发数据。数据是否有效，取决于应用软件。这就导致以下三种数据传输情形：

- 主器件发送数据——从器件发送无效 （Dummy）数据
- 主器件发送数据 —— 从器件发送数据
- 主器件发送无效数据 —— 从器件发送数据

图 19-2： SPI 主 / 从器件连接



19.3.5 主控模式

因为由主器件控制 SCKx 信号，所以它可以在任意时刻启动数据传输。主器件根据软件协议确定从器件（图 19-2 中的处理器 2）应在何时广播数据。

在主控模式下，数据一旦写入 SSPxBUF 寄存器就开始发送或接收。如果只打算将 SPI 作为接收器，则可以禁止 SDOx 输出（将其编程设置为输入）。SSPxSR 寄存器按设置的时钟速率，对 SDIx 引脚上的信号进行连续移位输入。每收到一个字节，就将其装入 SSPxBUF 寄存器，就像接收到普通字节一样（中断和状态位相应置 1）。这在以 “线路活动监控”（Line Activity Monitor）方式工作的接收器应用中很有用。

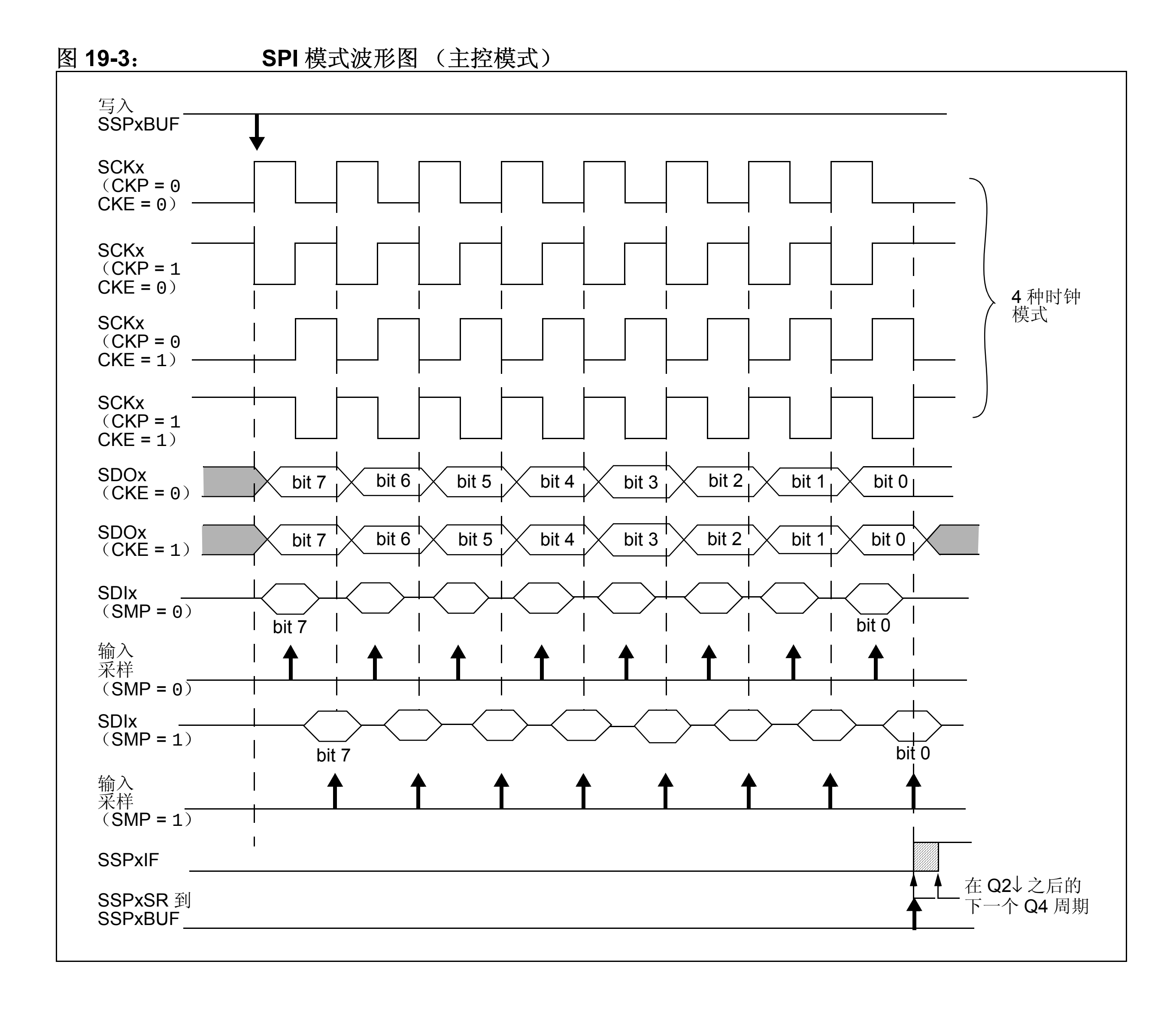
可通过对 CKP 位 （SSPxCON1<4>）进行适当的编程来选择时钟极性。图 19-3、图 19-5 和图 19-6 将给出 SPI 通信的时序图，其中最先发送的是最高有效位。在主控模式下， SPI 时钟速率（位速率）可由用户编程设定为下面几种之一：

- FOSC/4 （或 TCY）
- FOSC/16 （或 4 • TCY）
- FOSC/64 （或 16 • TCY）
- Timer2 输出 /2

这样可使数据速率最高达到 10.00 Mbps （时钟频率为 40 MHz）。

图 19-3 给出了主控模式的波形图。当 CKE 位置 1 时，SDOx 数据在 SCKx 出现时钟边沿前一直有效。图中所示的输入采样的变化由 SMP 状态位反映。图中给出了将接收到的数据装入 SSPxBUF 的时间。

图 19-3： SPI 模式波形图 （主控模式）



 超前信息

19.3.6 从动模式

在从动模式下，当 SCKx 引脚上有外部时钟脉冲时启动发送和接收数据。当最后一位数据被锁存后，中断标志位 SSPxIF 置 1。

在 SPI 从动模式下使能该模块时，时钟线必须与适当的空闲状态相匹配。时钟线可通过读 SCKx 引脚来查看。空闲状态由 CKP 位 （SSPxCON1<4>）决定。

在从动模式下，外部时钟由 SCKx 引脚上的外部时钟源提供。外部时钟必须满足电气规范中规定的高电平和低电平的最短时间要求。

在休眠模式下，从器件仍可发送 / 接收数据。当接收到一个字节时，器件从休眠状态中唤醒。

19.3.7 从动选择同步

SSx 引脚允许器件工作于同步从动模式。SPI 必须处于从动模式，并使能SSx引脚控制（SSPxCON1<3:0>=04h）。当 SSx 引脚为低电平时，使能数据的发送和接收，同时 SDOx 引脚被驱动。当 SSx 引脚变为高电平时，即使是在字节的发送过程中，也不再驱动 SDOx 引脚，而是将其变成悬空输出状态。根据应用需要，可在 SDOx 引脚上外接上拉 / 下拉电阻。

> 注 **1：** 当 SPI 处于从动模式，并且使能 SSx 引脚控制（SSPxCON1<3:0> = `0100`）时，如果 SSx 引脚设置为 VDD 电平将使 SPI 模块复位。
>
> **2：** 如果 SPI 工作在从动模式下并且 CKE 置 1，则必须使能 SSx 引脚控制。

当 SPI 模块复位后，位计数器被强制为 `0`。这是通过强制将SSx引脚拉为高电平或将SSPEN位清零来实现的。

将 SDOx 引脚和 SDIx 引脚相连，可以仿真二线制通信。当 SPI 需要作为接收器工作时，SDOx 引脚可以被配置为输入端。这样就禁止了从 SDOx 发送数据。因为 SDIx 不会引起总线冲突，所以可以一直将其保留为输入（SDIx 功能）。

图 19-4： 从动同步波形图

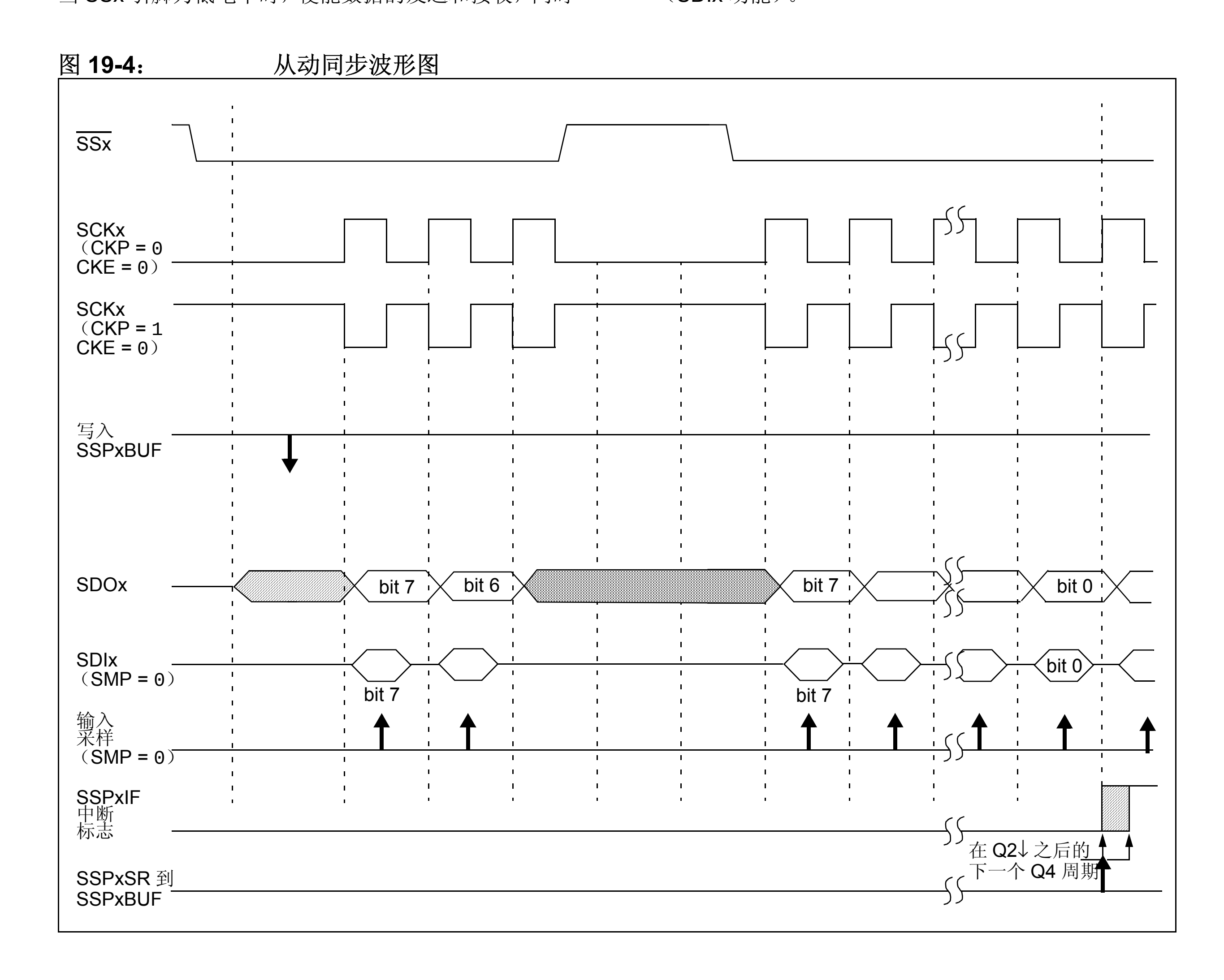
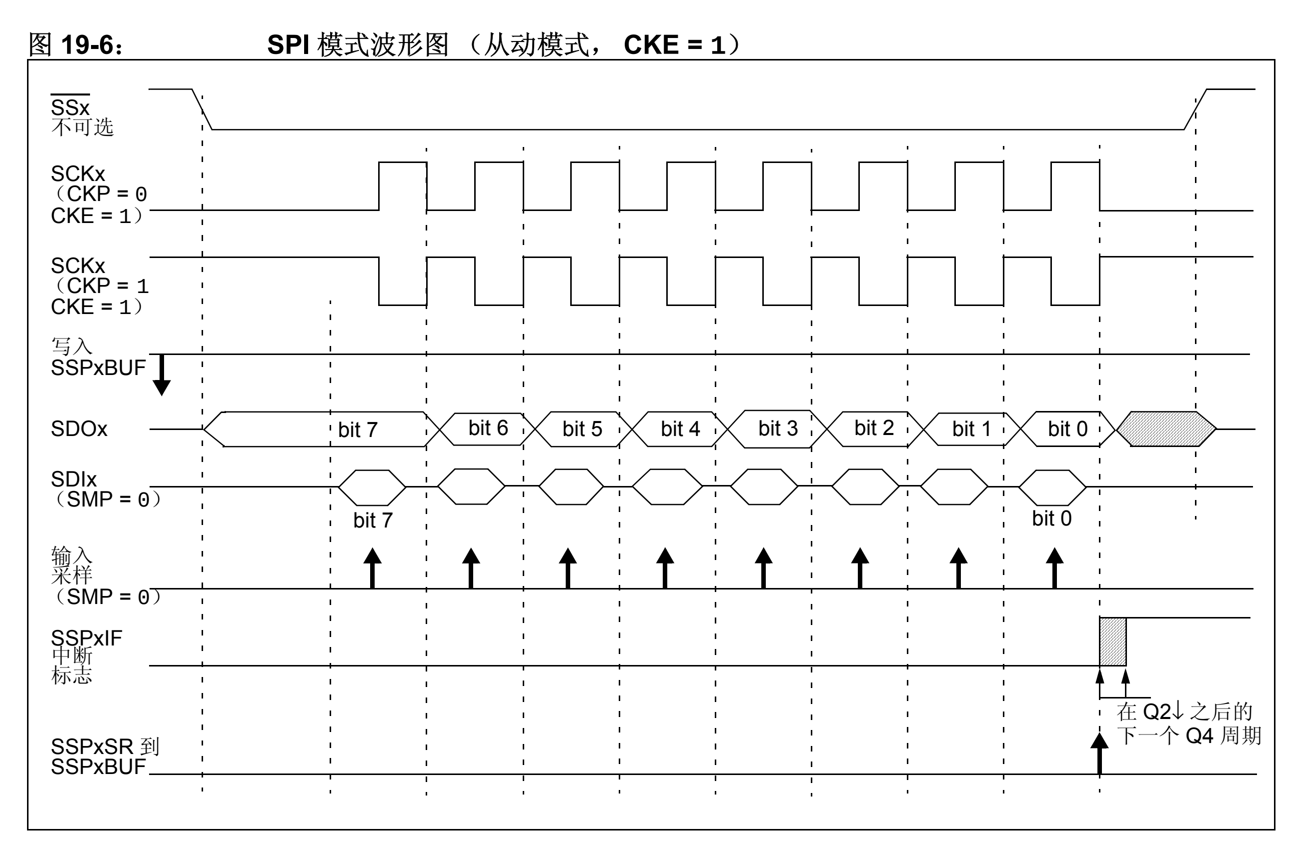


图 19-5: SPI 模式波形图 (从动模式, CKE = 0)

SSx
可选的
SCKx
(CKP = 0
CKE = 0)
SCKx
(CKP = 1
CKE = 0)
写入
SSPxBUF
SDOx
bit 7
bit 6
bit 5
bit 4
bit 3
bit 2
bit 1
bit 0
SDIx
(SMP = 0)
bit 7
bit 0
输入
采样
(SMP = 0)
SSPxIF
中断
标志
SSPxSR 到
SSPxBUF
在 Q2↓ 之后的
下一个 Q4 周期

图 19-6: SPI 模式波形图 (从动模式, CKE = 1)



超前信息

19.3.8 在功耗管理模式下的操作

在 SPI 主控模式下，模块时钟速度与全功耗模式下的不同；处于休眠模式时，所有时钟都停止。

在空闲模式下，需要为外设提供一个时钟。该时钟应该来自于主时钟源、辅助时钟源 （32.768 kHz 的 Timer1 振荡器）或 INTRC 时钟源。更多信息，请参见**第 2.7 节 “时钟源与振荡器切换”**。

在大多数情况下，主器件为 SPI 数据提供的时钟速度并不重要；但是，每个系统都应该评估此因素。

如果允许了 MSSP 中断，那么当主器件发送完数据时这些中断可以将控制器从休眠模式或某种空闲模式唤醒。如果不想从休眠或空闲模式退出，应该禁止 MSSP 中断。

如果选择了休眠模式，所有模块的时钟都将停止，并且在器件被唤醒前，发送 / 接收将保持此停滞状态。当器件返回到运行模式后，该模块将恢复发送和接收数据。

在 SPI 从动模式下， SPI 发送 / 接收移位寄存器与器件异步工作。这可以使器件处于任何功耗管理模式下，而且数据仍可被移入 SPI 发送 / 接收移位寄存器。当 8 位数据全部接收到后， MSSP 中断标志位将置 1，并且如果允许中断的话，器件被唤醒。

19.3.9 复位的影响

复位操作会禁止 MSSP 模块并终止当前的数据传输。

19.3.10 总线模式兼容性

表 19-1 中所示是标准 SPI 模式与 CKP 和 CKE 控制位状态的对应关系。

表 19-1： SPI 总线模式

标准 SPI 模式术语	控制位状态	
	CKP	**CKE**
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

还有一个 SMP 位用来控制数据何时被采样。

19.3.11 SPI 时钟速度和模块相互关系

因为 MSSP1 和 MSSP2 是独立的模块，它们可以使用不同的数据速率同时工作。设置 SSPxCON1 寄存器的 SSPM3:SSPM0 位来设定相应模块的速率。

有一种例外情况就是在两个模块都在主控模式下使用 Timer2 作为时基的时候。在这种情况下，任何对 Timer2 工作模式的更改都会对两个 MSSP 模块造成相同的影响。如果每个模块需要不同的位速率，用户应该为一个模块选择其他三种时基中的一种。

表 19-2： 与 SPI 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF(1)	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE(1)	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP(1)	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	61
TRISD	TRISD7(1)	TRISD6(1)	TRISD5(1)	TRISD4(1)	TRISD3	TRISD2	TRISD1	TRISD0	61
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	61
SSP1BUF	MSSP1 接收缓冲区 / 发送寄存器								60
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSP1STAT	SMP	CKE	$D/\overline{A}$	P	S	$R/\overline{W}$	UA	BF	60
SSP2BUF	MSSP2 接收缓冲区 / 发送寄存器								63
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	63
SSP2STAT	SMP	CKE	$D/\overline{A}$	P	S	$R/\overline{W}$	UA	BF	63

图注： SPI 模式下的 MSSP 模块不使用阴影单元。

注 1： 这些位仅在 100 引脚器件上实现；在其他器件上未实现且读为 0。

19.4 I²C 模式

MSSP 模块工作在 I²C 模式时，可以实现所有的主控和从动功能 （包括广播呼叫支持），并且硬件上提供启动位和停止位的中断来判断总线何时空闲 （多主机功能）。 MSSP 模块实现了标准模式规范以及 7 位和 10 位寻址。

有两个引脚用于数据传输：

- 串行时钟 （SCLx）——RC3/SCK1/SCL1 （或 RD6/SCK2/SCL2，适用于 100 引脚器件）
- 串行数据 （SDAx）——RC4/SDI1/SDA1 （或 RD5/SDI2/SDA2，适用于 100 引脚器件）

用户必须通过将 TRISC<4:3> 或 TRISD<5:4> 位置 1 将这些引脚配置为输入引脚。

图 19-7： MSSP 框图 （I²C™ 模式）

内部
数据总线
读
写
SSPxBUF 寄存器
SCLx
移位
时钟
SDAx
SSPxSR 寄存器
MSb
LSb
匹配检测
地址匹配
地址屏蔽
SSPxADD 寄存器
启动和
停止位检测
置 1 或复位
S 位和 P 位
（SSPxSTAT
寄存器）

19.4.1 寄存器

MSSP 模块有 6 个寄存器用于 I²C 操作。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPxCON1）
- MSSP 控制寄存器 2 （SSPxCON2）
- MSSP 状态寄存器 （SSPxSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPxBUF）
- MSSP 移位寄存器 （SSPxSR）——不可直接访问
- MSSP 地址寄存器 （SSPxADD）

SSPxCON1、SSPxCON2 和 SSPxSTAT 是在 I²C 模式下的控制寄存器和状态寄存器。 SSPxCON1 和 SSPxCON2 寄存器是可读写的。 SSPxSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPxCON2 的多数位会呈现不同的功能，取决于该模块是运行在主控还是从动模式下；SSPxCON2<5:1>位在从动模式下还具有不同的名称。寄存器 19-5 （主控模式）和寄存器 19-6 （从动模式）显示了 SSPxCON2 的不同方面。

SSPxSR 是用来将数据移入或移出的移位寄存器。 SSPxBUF 是缓冲寄存器，可用于数据字节的写入或读出。

当 MSSP 被配置为工作在 I²C 从动模式下时，SSPxADD 寄存器将保存从器件的地址。当 MSSP 工作在主控模式下时，SSPxADD 的低 7 位用作波特率发生器的重载值。

接收数据时，SSPxSR 和 SSPxBUF 共同构成一个双重缓冲接收器。当 SSPxSR 接收到一个完整的字节之后，该字节会被送入 SSPxBUF，同时将中断标志位 SSPxIF 置 1。

在数据发送过程中， SSPxBUF 不是双重缓冲的，对 SSPxBUF 的写操作将同时写入 SSPxBUF 和 SSPxSR。

寄存器 19-3： SSPxSTAT：MSSPx 状态寄存器 （I^2C™ 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P[1]	S[1]	R/$\overline{W}$[2,3]	UA	BF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SMP**：边沿斜率控制位
在主控或从动模式下：
1 = 标准速度模式下禁止边沿斜率控制 （100 kHz 和 1 MHz）
0 = 高速模式下使能边沿斜率控制 （400 kHz）

bit 6 **CKE**：SMBus 选择位
在主控或从动模式下：
1 = 使能 SMBus 特定输入
0 = 禁止 SMBus 特定输入

bit 5 **D/$\overline{A}$**：数据 / $\overline{地址}$位
在主控模式下：
保留。
在从动模式下：
1 = 表示上一个接收或发送的字节是数据
0 = 表示上一个接收或发送的字节是地址

bit 4 **P**：停止位 [1]
1 = 表示最近检测到停止位
0 = 最近未检测到停止位

bit 3 **S**：启动位 [1]
1 = 表示最近检测到起始位
0 = 最近未检测到起始位

bit 2 **R/$\overline{W}$**：读 / $\overline{写}$信息位 （仅用于 I^2C 模式） [2,3]
在从动模式下：
1 = 读
0 = 写
在主控模式下：
1 = 正在进行发送
0 = 未进行发送

bit 1 **UA**：更新地址位 （仅用于 10 位从动模式）
1 = 表示用户需要更新 SSPxADD 寄存器中的地址
0 = 不需要更新地址

bit 0 **BF**：缓冲区满状态位
在发送模式下：
1 = SSPxBUF 已满
0 = SSPxBUF 为空
在接收模式下：
1 = SSPxBUF 已满 （不包括 $\overline{ACK}$ 和停止位）
0 = SSPxBUF 为空 （不包括 ACK 和停止位）

注 1： 该位在复位及 SSPEN 清零时被清零。
2： 该位保存最后一个地址匹配后的 R/$\overline{W}$ 位信息。该位仅在从地址匹配到下一个起始位、停止位或非 $\overline{ACK}$ 位之间有效。
3： 将该位与 SEN、 RSEN、 PEN、 RCEN 或 ACKEN 进行或运算将指示 MSSP 是否处于有效模式。

寄存器 19-4： SSPxCON1：MSSPx 控制寄存器 1 （I²C™ 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **WCOL**：写冲突检测位

在主控发送模式下：
1 = 当 I²C 不满足启动发送数据的条件时，试图向 SSPxBUF 寄存器写入数据 （必须用软件清零）
0 = 未发生冲突

在从动发送模式下：
1 = 正在发送前一个字时，又有数据写入 SSPxBUF 寄存器 （必须用软件清零）
0 = 未发生冲突

在接收模式 （主控或从动模式）下：
该位是无关位。

bit 6 **SSPOV**：接收溢出指示位

在接收模式下：
1 = SSPxBUF 寄存器仍在保存前一字节时，接收到一个新的字节 （必须用软件清零）
0 = 无溢出

在发送模式下：
在发送模式下，该位是无关位。

bit 5 **SSPEN**：主控同步串口使能位

1 = 使能串口并将 SDAx 和 SCLx 引脚配置为串口引脚 (1)
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚 (1)

bit 4 **CKP**：SCKx 释放控制位

在从动模式下：
1 = 释放时钟
0 = 保持时钟低电平 （时钟延长），用来确保数据建立时间

在主控模式下：
在此模式下未使用。

bit 3-0 **SSPM3:SSPM0**：主控同步串口模式选择位

1111 = I²C 从动模式， 10 位地址，并使能启动位和停止位中断 (2)
1110 = I²C 从动模式， 7 位地址，并使能启动位和停止位中断 (2)
1011 = I²C 由固件控制的主控模式 （从器件空闲） (2)
1000 = I²C 主控模式，时钟 = FOSC/(4 * (SSPADD + 1))(2)
0111 = I²C 从动模式， 10 位地址 (2)
0110 = I²C 从动模式， 7 位地址 (2)

注 1： 当该位被使能时，必须将 SDAx 和 SCLx 引脚配置为输入引脚。
2： 此处未列出的位组合为保留或只用于 SPI 模式。

寄存器 19-5： SSPxCON2：MSSPx 控制寄存器 2 （I²C™ 主控模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT[1]	ACKEN[2]	RCEN[2]	PEN[2]	RSEN[2]	SEN[2]
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **GCEN**：广播呼叫使能位 （仅用于从动模式）
在主控模式下未使用。

bit 6 **ACKSTAT**：应答状态位 （仅用于主控发送模式）
1 = 未收到来自从器件的应答
0 = 收到来自从器件的应答

bit 5 **ACKDT**：应答数据位 （仅用于主控接收模式）[1]
1 = 无应答
0 = 应答

bit 4 **ACKEN**：应答序列使能位 [2]
1 = 在 SDAx 和 SCLx 引脚上发起应答序列，并发送 ACKDT 数据位。由硬件自动清零。
0 = 应答序列空闲

bit 3 **RCEN**：接收使能位 （仅用于主控接收模式）[2]
1 = 使能 I²C 接收模式
0 = 接收空闲

bit 2 **PEN**：停止条件使能位 [2]
1 = 在 SDAx 和 SCLx 引脚上发起停止条件。由硬件自动清零。
0 = 停止条件空闲

bit 1 **RSEN**：重复启动条件使能位 [2]
1 = 在 SDAx 和 SCLx 引脚上发起重复启动条件。由硬件自动清零。
0 = 重复启动条件空闲

bit 0 **SEN**：启动条件使能 / 延长使能位 [2]
1 = 在 SDAx 和 SCLx 引脚上发起启动条件。由硬件自动清零。
0 = 启动条件空闲

注 **1：** 用户在接收结束时发起一个应答时序，同时发送该值。
2： 如果 I²C 模块处于激活状态，可能这些位不会被置 1 （没有缓存），并且也可能不会写入 SSPxBUF （或禁止写 SSPxBUF）。

寄存器 19-6： SSPxCON2：MSSPx 控制寄存器 2 （I^2C™ 从动模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ADMSK5	ADMSK4	ADMSK3	ADMSK2	ADMSK1	SEN[(1)]
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **GCEN**：广播呼叫使能位 （仅用于从动模式）
1 = 当 SSPxSR 接收到广播呼叫地址 （0000h）时允许中断
0 = 禁止广播呼叫地址

bit 6 **ACKSTAT**：应答状态位
在从动模式下未使用。

bit 5-2 **ADMSK5:ADMSK2**：从动地址屏蔽选择位
1 = 使能 SSPxADD 相应位的屏蔽
0 = 禁止 SSPxADD 相应位的屏蔽

bit 1 **ADMSK1**：从动地址最低有效位屏蔽选择位
在 7 位地址模式下：
1 = 仅使能 SSPxADD<1> 屏蔽
0 = 禁止 SSPxADD<1> 屏蔽
在 10 位地址模式下：
1 = 使能 SSPxADD<1:0> 屏蔽
0 = 禁止 SSPxADD<1:0> 屏蔽

bit 0 **SEN**：延长使能位 [(1)]
1 = 为从动发送和从动接收使能时钟延长
0 = 时钟延长被禁止

注 1： 如果 I^2C 模块处于激活状态，可能该位不会被置 1 （没有缓存），并且也可能不会写入 SSPxBUF （或禁止写 SSPxBUF）。

19.4.2 工作原理

通过将 MSSP 使能位 SSPEN（SSPxCON1<5>）置 1，可使能 MSSP 模块。

SSPxCON1 寄存器用于控制 I^{2}C 工作模式。可通过设置模式选择位（SSPxCON1<3:0>）选择以下 I^{2}C 模式之一：

- I^{2}C 主控模式，时钟 = (Fosc/4) x (SSPxADD + 1)
- I^{2}C 从动模式（7 位地址）
- I^{2}C 从动模式（10 位地址）
- I^{2}C 从动模式（7 位地址），使能启动位和停止位中断
- I^{2}C 从动模式（10 位地址），使能启动位和停止位中断
- I^{2}C 固件控制的主控模式，从器件空闲

通过将相应的 TRISC 或 TRISD 位置 1，将 SCLx 和 SDAx 引脚编程为输入引脚；在 SSPEN 位置 1 时选择任何 I^{2}C 模式，将强制上述引脚漏极开路。要确保此模块的正常工作，必须为 SCLx 和 SDAx 引脚提供外接上拉电阻。

19.4.3 从动模式

在从动模式下， SCLx 引脚和 SDAx 引脚必须被配置为输入（TRISC<4:3> 或 TRISD<5:4> 置 1）。必要时 MSSP 模块将使用输出数据改写输入状态（从发送器）。

I^{2}C 从动模式硬件总是在地址精确匹配时产生中断。此外，地址屏蔽功能可使硬件在多个地址发生匹配时（7 位寻址模式下多达 31 个，10 位寻址模式下多达 63 个）产生一个中断。用户也可以通过模式选择位，选择使用启动位或停止位中断。

当地址匹配或在地址匹配后发送的数据被接收时，硬件会自动产生一个应答（ACK）脉冲，并把当时 SSPxSR 寄存器中接收到的值装入 SSPxBUF 寄存器。

只要满足下列条件之一，MSSP 模块就不会产生此 $\overline{\text{ACK}}$ 脉冲：

- 在接收到数据前，缓冲区满位 BF（SSPxSTAT<0>）被置 1。
- 在接收到数据前，溢出位 SSPOV（SSPxCON1<6>）被置 1。

在上述情况下，SSPxSR寄存器的值不会装入SSPxBUF，但是 SSPxIF 位会置 1。BF 位是通过读取 SSPxBUF 寄存器清零的，而 SSPOV 位是通过软件清零的。

为确保正常工作， SCLx 时钟输入必须满足最小高电平和最小低电平时间要求。在时序参数 100 和参数 101 中显示了 I^{2}C 规范的高低电平时间和对 MSSP 模块的具体要求。

19.4.3.1 寻址

一旦 MSSP 模块被使能，它就会等待启动条件出现。启动条件出现后，8 位数据被移入 SSPxSR 寄存器。在时钟（SCLx）线的上升沿采样所有的输入位。寄存器 SSPxSR<7:1> 的值会和 SSPxADD 寄存器的值比较，该比较是在第 8 个时钟（SCLx）脉冲下降沿进行的。如果地址匹配，并且 BF 位和 SSPOV 位都被清零，会发生下列事件：

1. SSPxSR 寄存器值被装入 SSPxBUF 寄存器。
2. 将缓冲区满标志位 BF 置 1。
3. 产生 $\overline{\text{ACK}}$ 脉冲。
4. 在第 9 个 SCLx 脉冲的下降沿，MSSP 中断标志位 SSPxIF 置 1（如果允许中断，则产生中断）。

在 10 位地址模式下，从器件需要接收两个地址字节。第一个地址字节的高 5 位将指定这是否是一个 10 位地址。R/$\overline{\text{W}}$ 位（SSPxSTAT<2>）必须指定写操作，这样从器件才能接收到第二个地址字节。对于 10 位地址，第一个字节应该是“`11110 A9 A8 0`”，其中“A9”和“A8”是该地址的两个最高有效位。10 位地址模式的操作步骤如下，其中 7-9 步是针对从发送器而言的。

1. 接收地址的第一个（高）字节（SSPxIF、BF 和 UA 位置 1）。
2. 用地址的第二个（低）字节更新 SSPxADD 寄存器（UA 位清零并释放 SCLx 时钟线）。
3. 读 SSPxBUF 寄存器（BF 位清零）并将标志位 SSPxIF 清零。
4. 接收地址的第二个（低）字节（SSPxIF、BF 和 UA 位置 1）。
5. 使用地址的第一个（高）字节更新 SSPxADD 寄存器。如果匹配的话就释放 SCLx 时钟线，这将清零 UA 位。
6. 读 SSPxBUF 寄存器（BF 位清零）并将标志位 SSPxIF 清零。
7. 接收重复启动条件。
8. 接收地址的第一个（高）字节（SSPxIF 位和 BF 位置 1）。
9. 读 SSPxBUF 寄存器（BF 位清零）并将标志位 SSPxIF 清零。

19.4.3.2 地址屏蔽

将地址的某一位屏蔽意味着该位可为任意值，此时会响应两个地址并产生一个中断。由于同一时刻可以有多个地址位被屏蔽，所以在 7 位模式下可响应多达 31 个地址，而在 10 位模式下则可响应多达 63 个地址（见例 19-2）。

不管是否使用地址屏蔽，I²C 从器件的工作方式保持不变。当使用地址屏蔽时，I²C 从器件能够响应多个地址并产生中断，此时需要通过查询 SSPxBUF 来判断是哪一个地址引起的中断。

在 7 位地址模式下，地址屏蔽位 ADMSK<5:1>（SSPxCON2<5:1>）可用来屏蔽 SSPxADD 寄存器中对应的地址位。如果ADMSK的某位被置1（ADMSK<n> = 1），则对应的地址位可以被忽略（SSPxADD<n> = x）。对于发出地址应答的模块来讲，只要与没被屏蔽的地址位匹配就可以了。

在 10 位地址模式下，地址屏蔽位 ADMSK<5:2> 可用来屏蔽 SSPxADD 寄存器中对应的地址位，而 ADMSK1 可以同时屏蔽地址的低 2 位（SSPxADD<1:0>）。如果 ADMSK 的某位是有效的（ADMSK<n> = 1），则对应的地址位可以被忽略（SSPxADD<n> = x）。需要注意的是，尽管在 10 位地址模式下，地址的高位也要用到 SSPxADD 寄存器中的某些位，但地址屏蔽位对这些位不起作用，地址屏蔽位只会影响地址低字节中的位。

注	**1：** ADMSK1 屏蔽地址的低 2 位。
	2： 地址屏蔽不会对地址的高 2 位起作用。

例 19-2： 地址屏蔽示例

7 位寻址模式：

SSPxADD<7:1> = A0h （1010000）（SSPxADD<0> 假设为 0）

ADMSK<5:1> = 00111

可被应答的地址：A0h, A2h, A4h, A6h, A8h, AAh, ACh, AEh

10 位寻址模式：

SSPxADD<7:0> = A0h （10100000）（此例中地址高 2 位被忽略，因为它们不受影响）

ADMSK<5:1> = 00111

可被应答的地址：A0h, A1h, A2h, A3h, A4h, A5h, A6h, A7h, A8h, A9h, AAh, ABh, ACh, ADh, AEh, AFh

19.4.3.3 接收

当地址字节的 $R/\overline{W}$ 位清零并发生地址匹配时，SSPxSTAT 寄存器的 $R/\overline{W}$ 位清零。接收的地址被装入 SSPxBUF 寄存器，且 SDAx 信号保持低电平（$\overline{ACK}$）。

当发生地址字节溢出时，则不会产生应答脉冲（$\overline{ACK}$）。溢出条件是指 BF 位（SSPxSTAT<0>）置 1，或者 SSPOV 位（SSPxCON1<6>）置 1。

每个数据传输字节都会产生一个 MSSP 中断。中断标志位 SSPxIF 必须用软件清零。通过 SSPxSTAT 寄存器可以确定该字节的状态。

如果 SEN 被使能（SSPxCON2<0> = 1），SCKx/SCLx（RC3 或 RD6）将在每个数据传输之后保持为低电平（时钟延长）。必须通过将 CKP 位（SSPxCON1<4>）置 1 才能释放时钟。更多详细信息，请参见**第 19.4.4 节“时钟延长”**。

19.4.3.4 发送

当输入的地址字节的 $R/\overline{W}$ 位置 1 并发生地址匹配时，SSPxSTAT 寄存器的 $R/\overline{W}$ 位置 1。接收到的地址被装入 SSPxBUF 寄存器。$\overline{ACK}$ 脉冲在第 9 位上发送，同时不管 SEN 的值如何，RC3 或 RD6 引脚保持低电平（更多详细信息，请参见**第 19.4.4 节“时钟延长”**）。通过延长时钟，主器件只有在从器件准备好发送数据时，才发出另一个时钟脉冲。发送的数据必须被装入 SSPxBUF 寄存器，同时也被装入 SSPxSR 寄存器。然后，应该通过将 CKP（SSPxCON1<4>）置 1 来使能 RC3 或 RD6 引脚。8 个数据位在 SCLx 输入的下降沿被移出。这可确保在 SCLx 为高电平期间 SDAx 信号是有效的（图 19-10）。

来自主接收器的 $\overline{ACK}$ 脉冲将在第 9 个 SCLx 输入脉冲的上升沿锁存。如果 SDAx 信号为高电平（无 $\overline{ACK}$ 应答信号），那么表示数据传输已完成。在这种情况下，如果从器件锁存了 $\overline{ACK}$，将复位从动逻辑（复位 SSPxSTAT寄存器），同时从器件监视下一个起始位的出现。如果 SDAx 线为低电平（$\overline{ACK}$），则必须将下一个要发送的数据装入 SSPxBUF 寄存器。同样，必须通过将 CKP 位置 1 来使能 RC3 或 RD6 引脚。

每传输一个数据字节都会产生一个MSSP中断。SSPxIF 位必须用软件清零，SSPxSTAT 寄存器用于确定字节的状态。SSPxIF 位在第 9 个时钟脉冲的下降沿被置 1。

图 19-8： **I²C™ 从动模式接收时序 （SEN = 0， 7 位地址）**

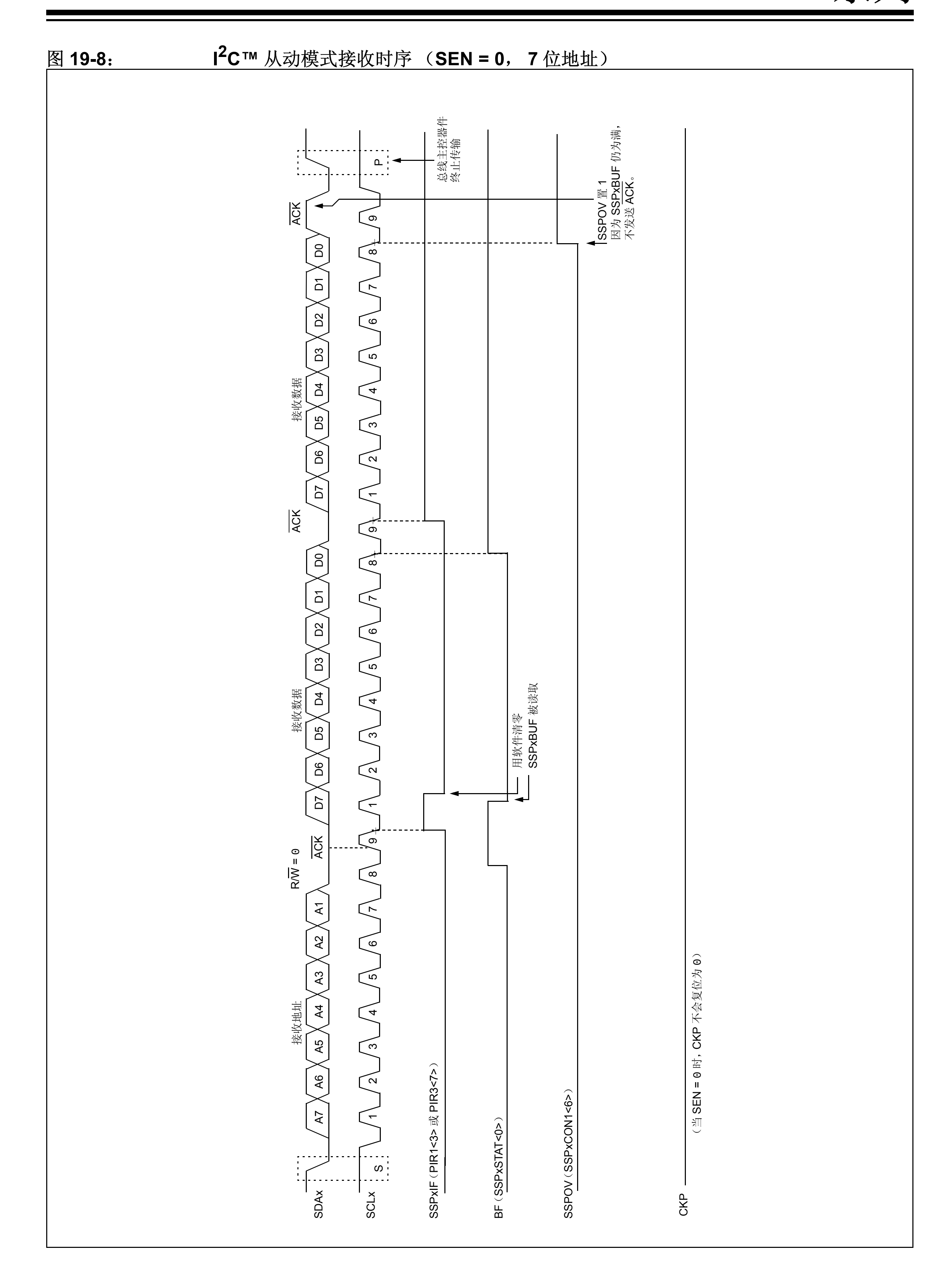


图 19-9：　　I²C™ 从动模式接收时序（SEN = 0 且 ADMSK<5:1> = 01011， 7 位地址）

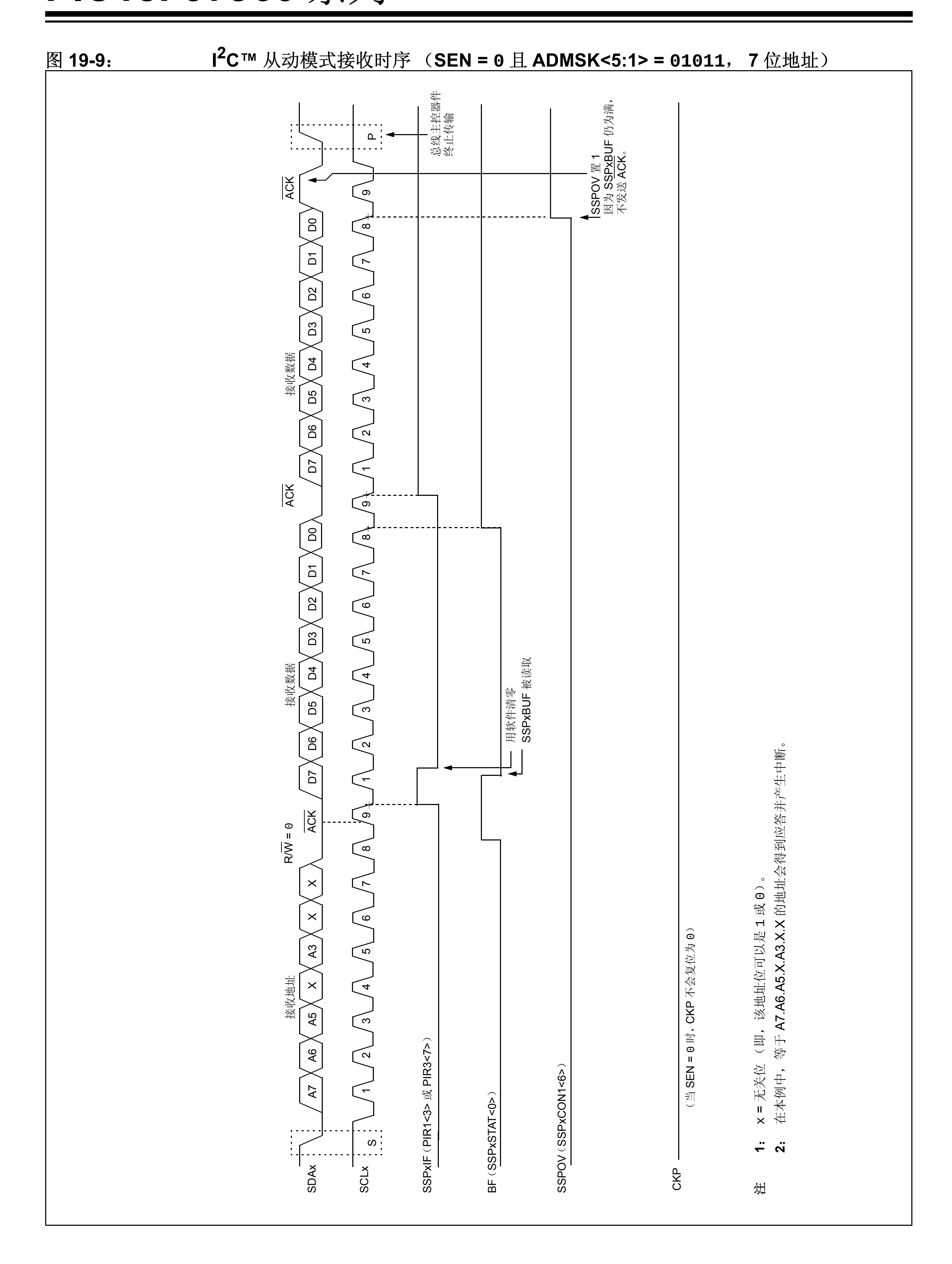


图 19-10： I^2C™ 从动模式发送时序 （7 位地址）

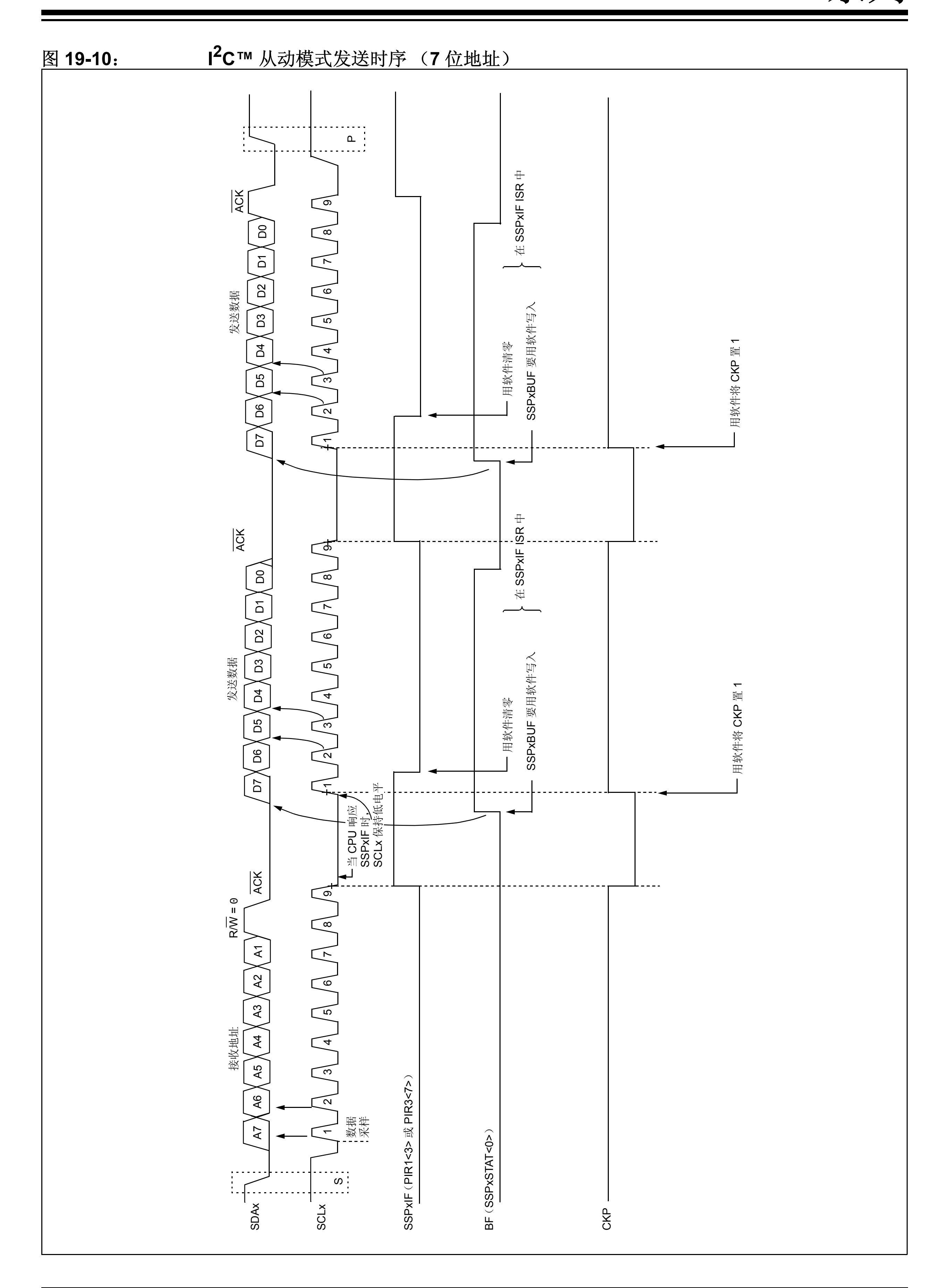
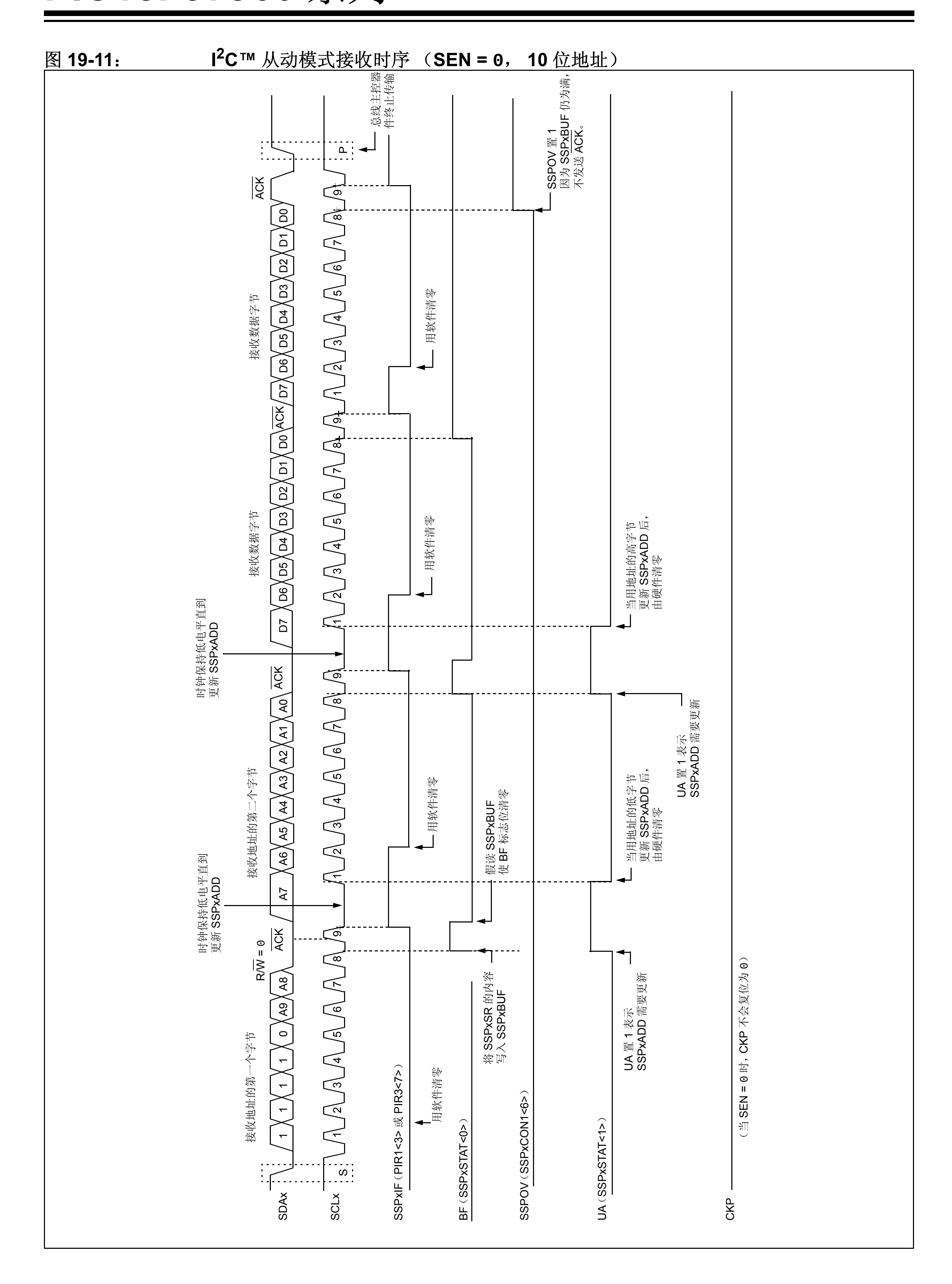


图 19-11： I^2C™ 从动模式接收时序 （SEN = 0， 10 位地址）



　　超前信息　　

图 19-12：　I²C™ 从动模式接收时序（SEN = 0 且 ADMSK<5:1> = 01001， 10 位地址）

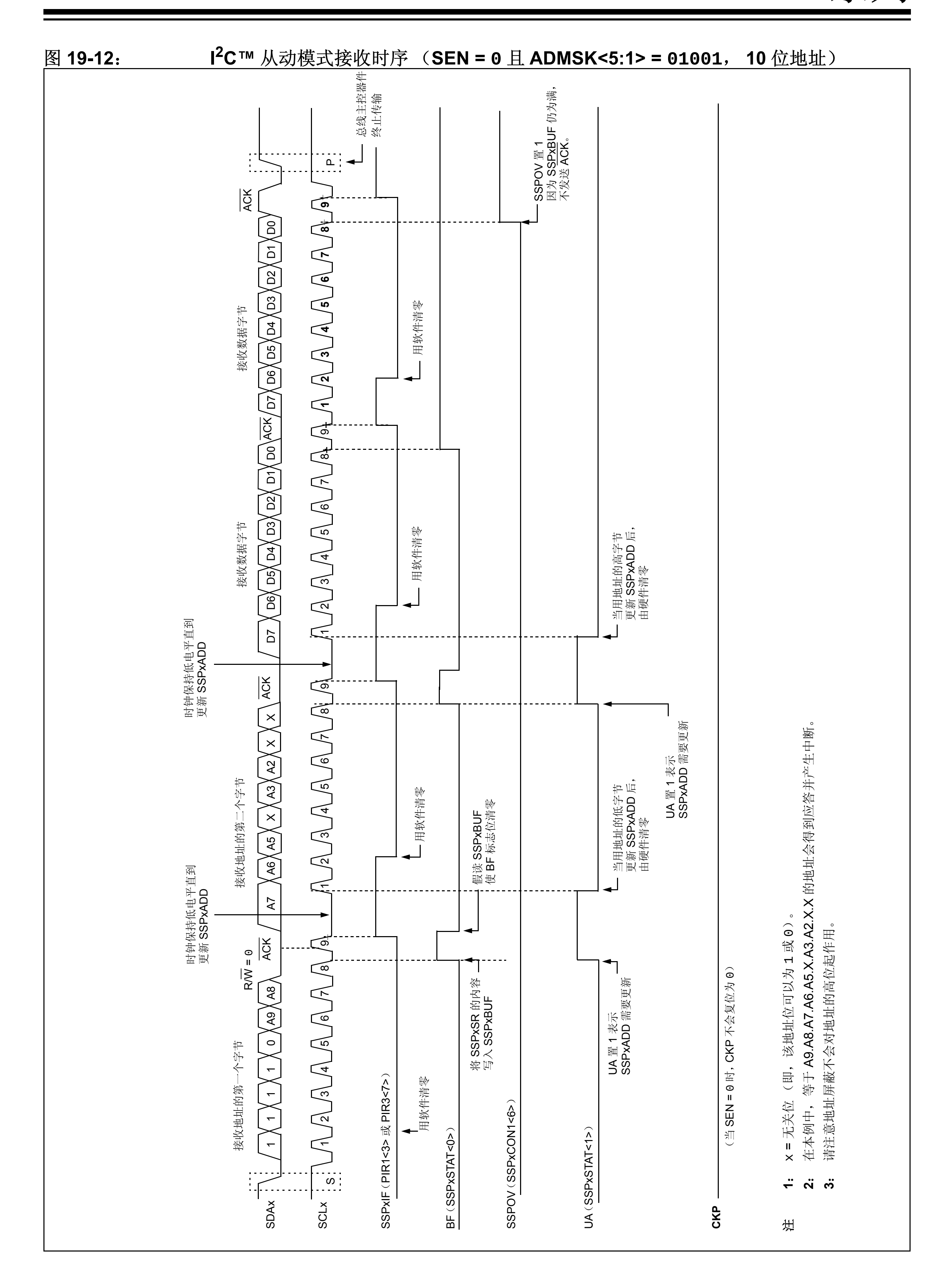
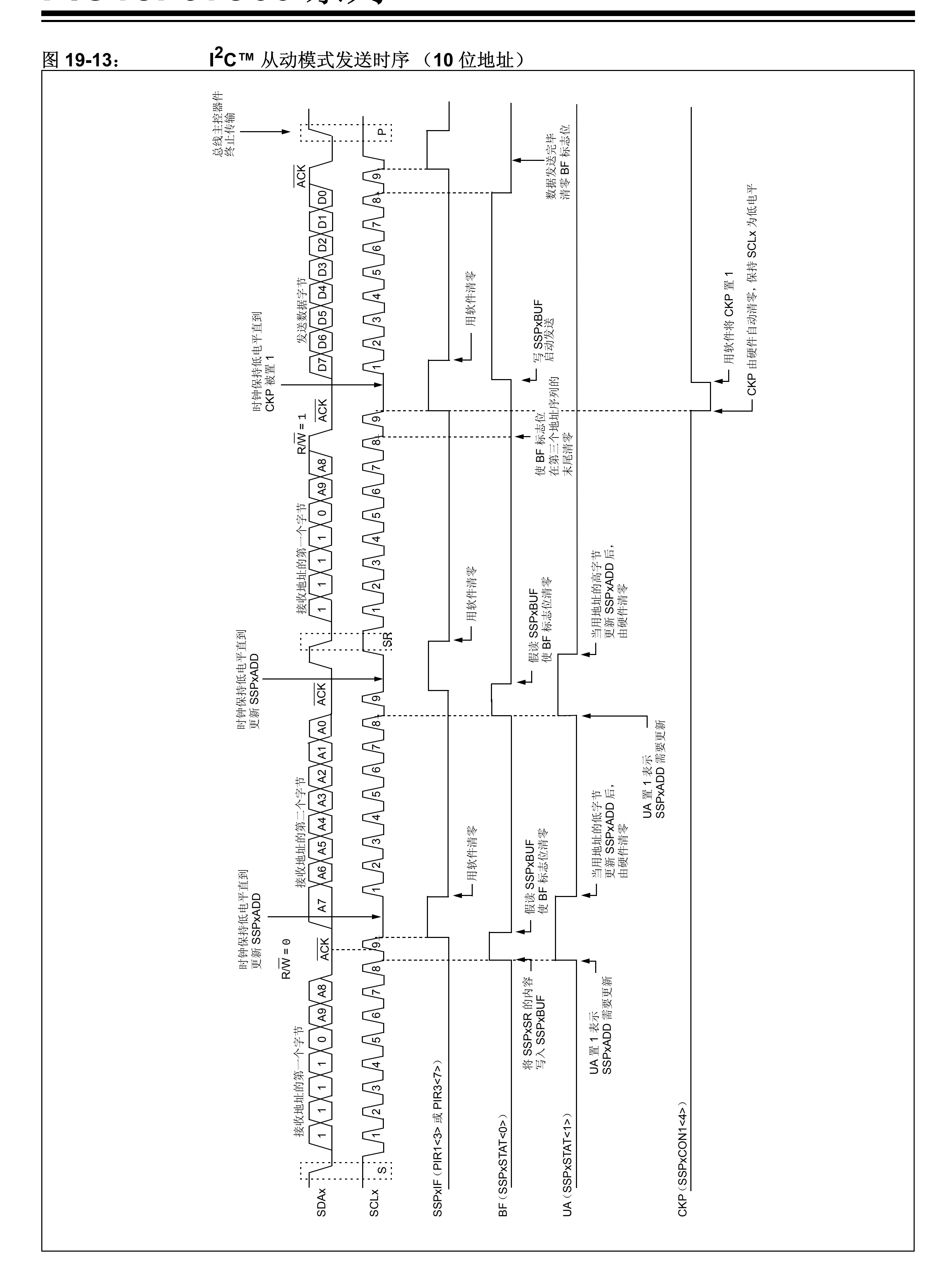


图 19-13： I²C™ 从动模式发送时序 （10 位地址）



 超前信息

19.4.4 时钟延长

7位和10位从动模式均能在发送序列期间自动实现时钟延长。

SEN 位（SSPxCON2<0>）允许在接收期间使能时钟延长。将 SEN 置 1 将使 SCLx 引脚在每个数据接收序列的末尾保持低电平。

19.4.4.1 7 位从动接收模式（SEN = 1）的时钟延长

在 7 位从动接收模式下，如果在 $\overline{\text{ACK}}$ 序列末的第 9 个时钟的下降沿将 BF 位置 1，则 SSPxCON1 寄存器中的 CKP 位就会自动清零，强制 SCLx 输出保持在低电平。CKP 被清零会将 SCLx 线拉为低电平。在允许继续接收之前，必须在用户的中断服务程序中将 CKP 位置 1。保持 SCLx 信号为低电平，用户可以在主器件发起另一个接收序列之前，有时间处理中断服务程序并读取 SSPxBUF 的内容。这将防止发生缓冲区溢出（见图 19-15）。

> **注 1：** 如果用户在第 9 个时钟的下降沿到来之前读取了 SSPxBUF 的内容，使得 BF 位被清零，那么 CKP 位就不会被清零，也不会发生时钟延长。
>
> **2：** 不管 BF 位的状态如何， CKP 位都可以用软件置 1。为避免溢出，在下一个接收序列开始之前，用户要注意在中断服务程序中清零 BF 位。

19.4.4.2 10 位从动接收模式（SEN = 1）的时钟延长

在 10 位从动接收模式下，在地址序列中会自动发生时钟延长，但是 CKP 位不会被清零。在这期间，如果 UA 位在第 9 个时钟之后置 1，将启动时钟延长。UA 位在接收到 10 位地址的高字节后被置 1，然后接收 10 位地址的第二个字节并清零 R/$\overline{\text{W}}$ 位。在更新 SSPxADD 的时候释放时钟线。如同 7 位模式一样，在每个数据接收序列中会发生时钟延长。

> **注：** 如果用户在第9个时钟的下降沿出现之前查询 UA 位，并通过更新 SSPxADD 寄存器清零 UA 位，而且在此之前用户没有读取 SSPxBUF 寄存器使 BF 位清零，则 CKP 位的电平仍然不会被拉低。基于 BF 位状态的时钟延长仅在数据序列中出现，不会出现在地址序列中。

19.4.4.3 7 位从动发送模式的时钟延长

如果 BF 位被清零， 7 位从动发送模式将通过在第 9 个时钟的下降沿出现后清零 CKP 位，以实现时钟延长。上述情形与 SEN 位的状态无关。

用户的中断服务程序必须先将CKP位置1才可以继续发送。在保持 SCLx 信号为低电平期间，用户在主器件发起另一个发送序列之前，将有时间处理中断服务程序并装入 SSPxBUF 的内容（见图 19-10）。

> **注 1：** 如果用户在第 9 个时钟的下降沿之前就装入 SSPxBUF 的内容，使 BF 位置 1，CKP 位就不会被清零，也不会发生时钟延长。
>
> **2：** 不管 BF 位的状态如何， CKP 位都可以用软件置 1。

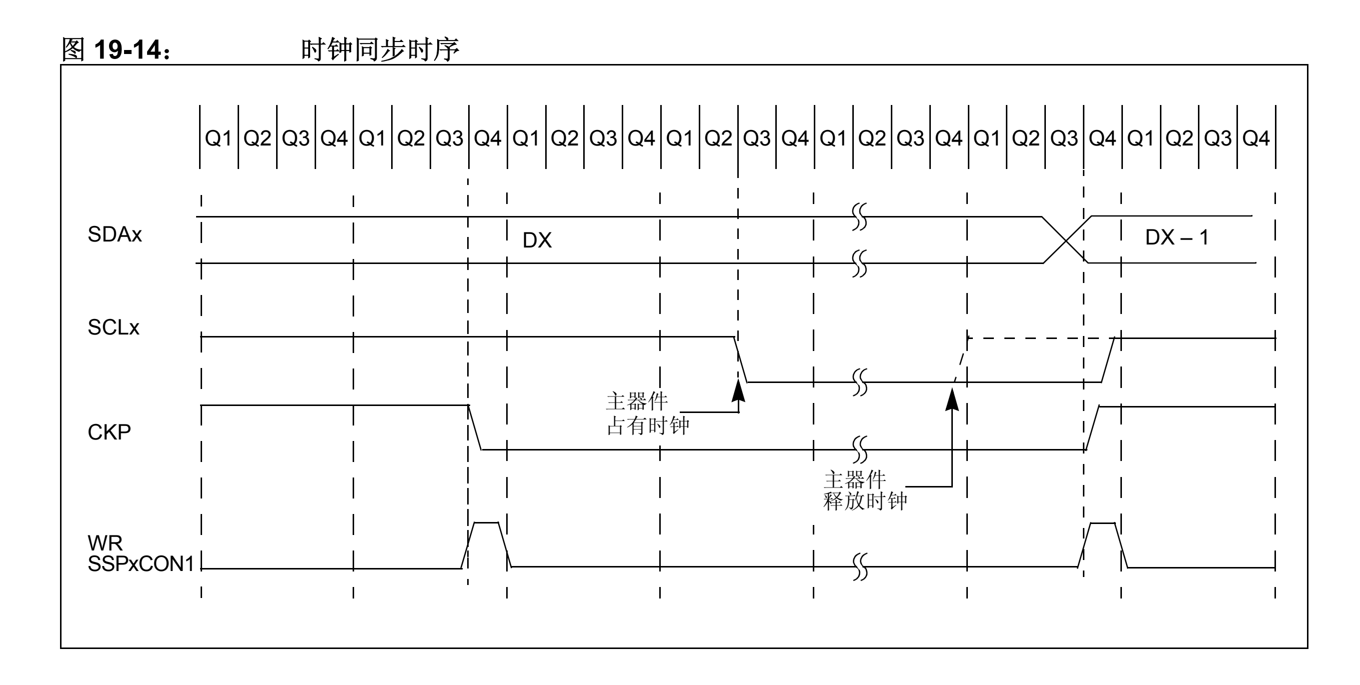
19.4.4.4 10 位从动发送模式的时钟延长

在 10 位从动发送模式下，在前两个地址序列中由 UA 位的状态来控制时钟延长，正如同 10 位从动接收模式一样。头两个地址后跟着第三个地址序列，该地址序列包含 10 位地址的高位和被置为 1 的 R/$\overline{\text{W}}$ 位。在执行完第三个地址序列后，UA 位不置 1，此时模块配置为发送模式，由 BF 标志位控制时钟延长，正如 7 位从动发送模式一样（见图 19-13）。

19.4.4.5 时钟同步和 CKP 位

当 CKP 位被清零时，SCLx 输出被强制为 0。然而，将 CKP 位清零不会将 SCLx 输出拉为低电平，除非已经采样到 SCLx 输出为低电平。因此，CKP 位不会将 SCLx 信号拉为低电平，除非外部 I^2C 主器件将 SCLx 线拉低。SCLx 输出将保持低电平，直到 CKP 位置 1 且 I^2C 总线上的其他器件将 SCLx 电平拉高为止。这可以确保对 CKP 位的写操作不会违反 SCLx 的最小高电平时间要求（见图 19-14）。

图 19-14： 时钟同步时序



超前信息

图 19-15： I²C™ 从动模式接收时序 （SEN = 1， 7 位地址）

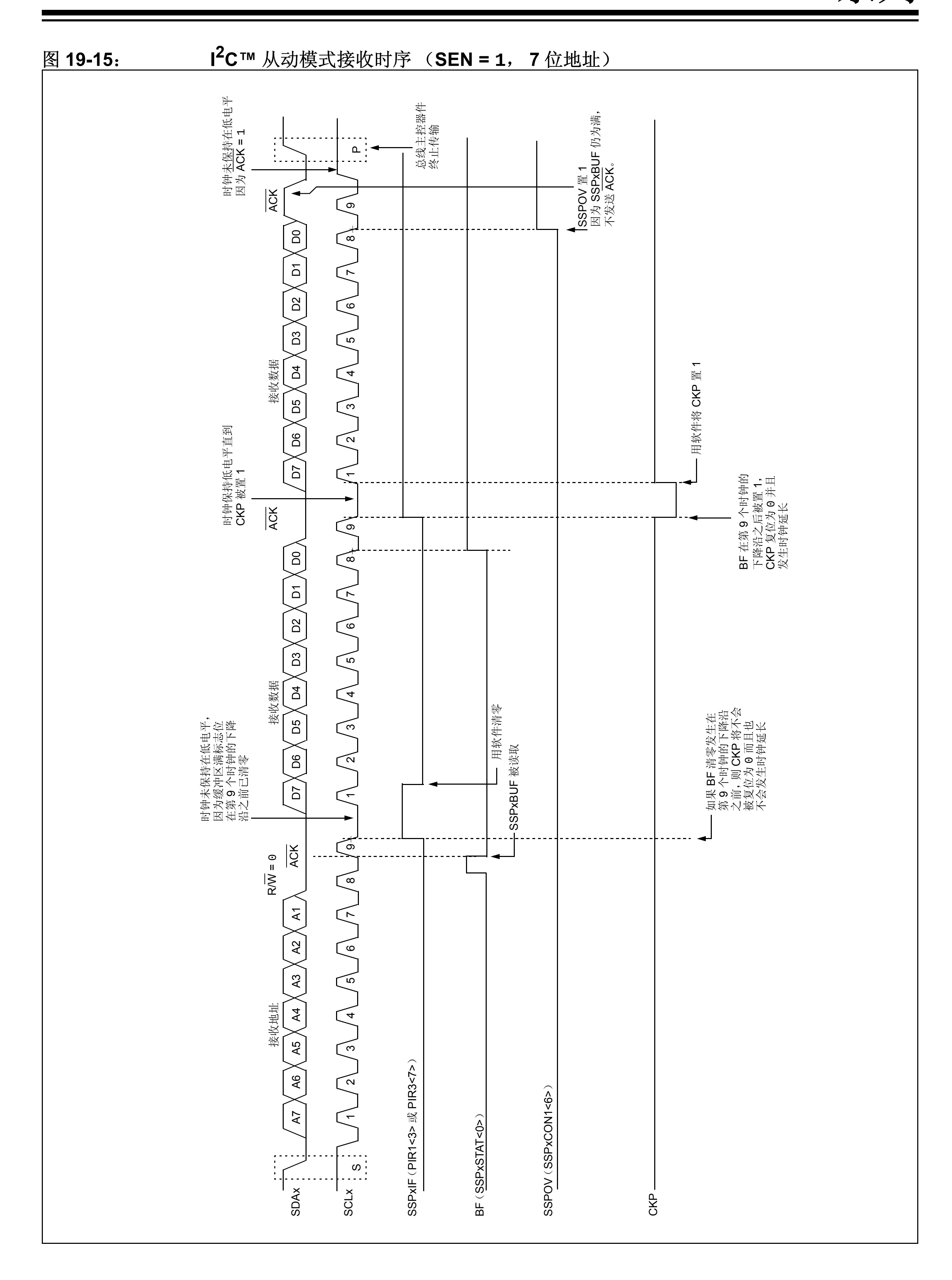
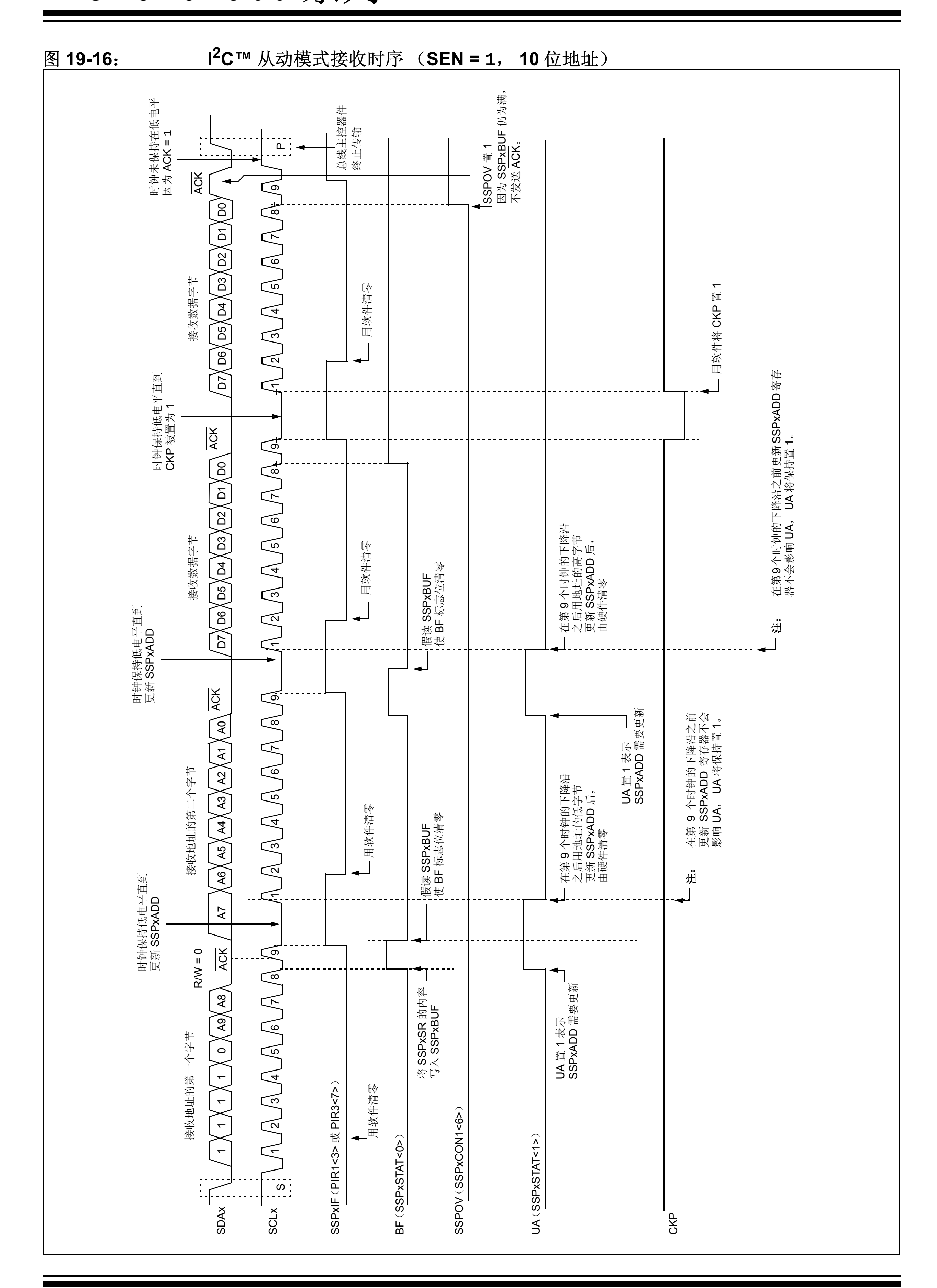


图 19-16： I²C™ 从动模式接收时序 （SEN = 1， 10 位地址）



 超前信息

19.4.5 支持广播呼叫地址

在 I²C 总线的寻址过程中，通常由启动条件后的第一个字节决定主器件将寻址哪个从器件。但广播呼叫地址例外，它能寻址所有器件。当使用这个地址时，理论上所有的器件都应该发送一个应答信号来响应。

广播呼叫地址是由 I²C 协议为特定目的保留的 8 个地址之一。它由全 0 组成，且 R/W = 0。

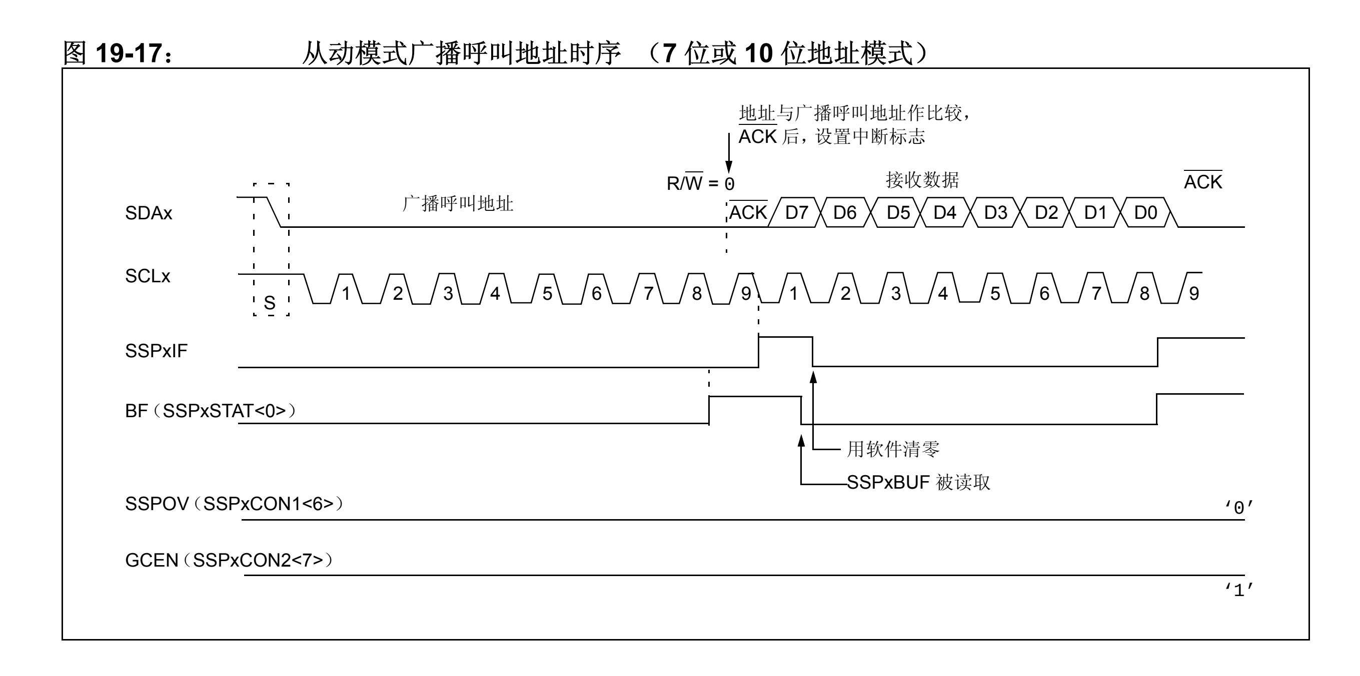
当使能广播呼叫使能位（GCEN）（SSPxCON2<7> 置 1）时，即可识别广播呼叫地址。检测到起始位后，8 位数据会被移入 SSPxSR，同时将该地址与 SSPxADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

如果与广播呼叫地址匹配，SSPxSR 的值将被传输到 SSPxBUF，BF 标志位（第 8 位）置 1，并且 SSPxIF 中断标志位在第 9 位（ACK 位）的下降沿置 1。

当中断得到响应时，可以通过读取 SSPxBUF 的内容来检查中断源。该值可用于判断是特定器件的地址还是一个广播呼叫地址。

在 10 位地址模式下，需要更新 SSPxADD 用来匹配地址的后半部分，同时 UA 位置 1（SSPxSTAT<1>）。如果 GCEN 位置 1 时采样到广播呼叫地址，同时从器件被配置为 10 位地址模式，则不再需要地址的后半部分，也不会将 UA 位置 1，从器件将在应答后开始接收数据（图 19-17）。

图 19-17：　从动模式广播呼叫地址时序　（7 位或 10 位地址模式）



19.4.6 主控模式

通过将 SSPxCON1 中的相应 SSPM 位置 1 和清零，同时将 SSPEN 位置 1，可以使能主控模式。在主控模式下， SCLx 和 SDAx 信号线由 MSSP 硬件控制。

主控模式通过在检测到启动和停止条件时产生中断来工作。停止 （P）位和启动 （S）位在复位或禁止 MSSP 模块时清零。当 P 位置 1 时，可以获得 I^2C 总线的控制权；否则， P 位和 S 位都清零，总线处于空闲状态。

在固件控制的主控模式下，用户代码根据启动和停止位条件执行所有的 I^2C 总线操作。

一旦使能主控模式，用户即可选择以下 6 项操作：

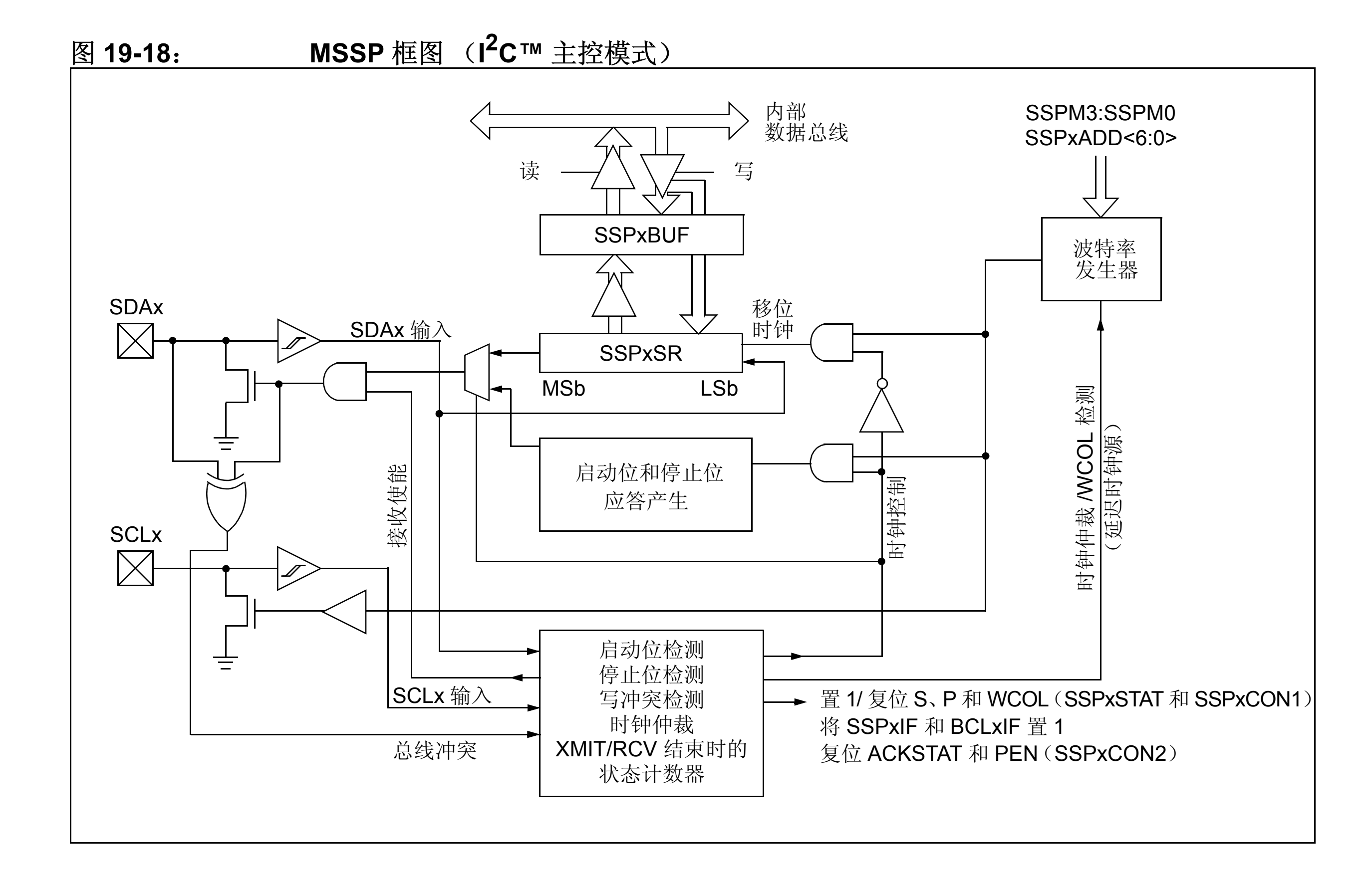
1. 在 SDAx 和 SCLx 上发出一个启动条件。
2. 在 SDAx 和 SCLx 上发出一个重复启动条件。
3. 写入 SSPxBUF 寄存器，启动数据 / 地址的发送。
4. 配置 I^2C 端口用于接收数据。
5. 在接收数据字节末尾产生应答信号。
6. 在 SDAx 和 SCLx 上产生停止条件。

注：	当配置为 I^2C 主控模式时，MSSP 模块不允许事件排队。例如，在启动条件结束前，不允许用户立即写SSPxBUF寄存器以启动传输。在这种情况下，将不会执行写 SSPxBUF， WCOL 位将被置 1，这表明没有发生对 SSPxBUF 的写操作。

下列事件会使 MSSP 中断标志位 SSPxIF 置 1（如果使能 MSSP 中断，则产生中断）：

- 启动条件
- 停止条件
- 数据字节发送 / 接收
- 应答发送
- 重复启动

图 19-18： MSSP 框图 （I^2C™ 主控模式）



 超前信息

19.4.6.1 I^2C 主控模式工作原理

主器件产生所有串行时钟脉冲和启动 / 停止条件。以停止条件或重复启动条件结束传输过程。因为重复启动条件也是下一次串行传输的开始，因此 I^2C 总线一直保持不被释放的状态。

在主控发送器模式下，串行数据通过 SDAx 输出，而串行时钟由 SCLx 输出。发送的第一个字节包括作为接收方的从器件地址 （7 位）和读 / 写 （R/$\overline{W}$）位。在这种情况下， R/$\overline{W}$ 位将是逻辑 `0`。一次发送 8 位串行数据。每发送一个字节，会收到一个应答位。输出启动和停止条件，表明串行传输的开始和结束。

在主控接收模式下，发送的第一个字节包括作为发送方的从器件地址（7 位）和 R/$\overline{W}$ 位。在这种情况下，R/$\overline{W}$ 将是逻辑 `1`。因此，发送的第一个字节是一个 7 位从器件地址，后面跟 `1` 表示接收。串行数据通过 SDAx 接收，而串行时钟由 SCLx 输出。每次接收 8 位串行数据。每接收到一个字节，都会发送一个应答位。启动和停止条件分别代表发送的开始和结束。

在 I^2C 模式下，将使用 SPI 模式中的波特率发生器将 SCLx 时钟频率设置为 100 kHz、 400 kHz 或 1 MHz。更多详细信息，请参见**第 19.4.7 节 "波特率"**。

下面是一个典型的发送时序：

1. 用户通过将启动使能位 SEN（SSPxCON2<0>）置 1，产生启动条件。
2. SSPxIF 置 1。在进行下一步操作前， MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPxBUF 进行发送。
4. 器件地址从 SDAx 引脚移出，直到发送完所有 8 位地址数据。
5. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPxCON2 寄存器（SSPxCON2<6>）。
6. MSSP 模块在第 9 个时钟周期的末尾将 SSPxIF 置 1，产生一个中断。
7. 用户将 8 位数据装入 SSPxBUF。
8. 数据从 SDAx 引脚移出，直到发送完所有 8 位数据。
9. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPxCON2 寄存器（SSPxCON2<6>）。
10. MSSP 模块在第 9 个时钟周期的末尾将 SSPxIF 置 1，产生一个中断。
11. 用户通过将停止使能位 PEN（SSPxCON2<2>）置 1 产生停止条件。
12. 一旦停止条件完成，将产生一个中断。

19.4.7 波特率

在 I^2C 主控模式下，波特率发生器（Baud Rate Generator，BRG）的重载值位于 SSPxADD 寄存器的低 7 位（图 19-19）。当发生对 SSPxBUF 的写操作时，波特率发生器将自动开始计数。BRG 会递减计数至 0，然后停止直到再次发生重载。BRG 计数器会在每个指令周期（TCY）中的 Q2 和 Q4 时钟周期上进行两次减计数。在 I^2C 主控模式下，会自动重载 BRG。

如果指定操作完成（即，在传输的最后一个数据位后面跟着 ACK），内部时钟将自动停止计数，SCLx 引脚将保持在其最后的状态。

表 19-3 给出了不同的指令周期下的时钟频率以及装入 SSPxADD 的 BRG 值。

19.4.7.1 波特率和模块的相互关系

因为 MSSP1 和 MSSP2 是彼此独立的模块，所以它们可以以不同的波特率同时在 I^2C 主控模式下工作。这是通过每个模块使用不同的 BRG 重载值实现的。

由于此模式的基本时钟源来自系统时钟，对系统时钟的任何更改将会同等程度地影响这两个模块。通过更改 BRG 重载值可以将一个或两个波特率改回到前一个值。



图 19-19： 波特率发生器框图

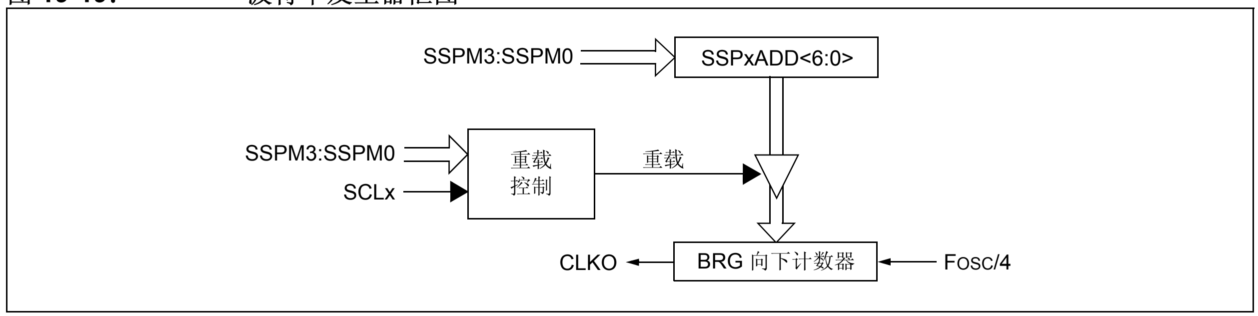


表 19-3： 使用 **BRG** 的 $\mathbf{I^2C}$**™** 时钟频率

Fosc	BRG 值	FSCL（两次 BRG 计满返回）
41.667 MHz	19h	400 kHz(1)
41.667 MHz	67h	100 kHz
31.25 MHz	13h	400 kHz(1)
31.25 MHz	4Dh	100 kHz
20.833 MHz	09h	400 kHz(1)
20.833 MHz	33h	100 kHz

注 1： 虽然 I^2C™ 接口各方面都不符合 400 kHz I^2C 规范（该规范适用于大于 100 kHz 的频率），但在需要较高频率的应用场合可以慎重使用。

超前信息

19.4.7.2　时钟伸裁

如果在任何接收、发送或重复启动 / 停止条件过程中，主器件拉高了 SCLx 引脚（允许 SCLx 引脚悬空为高电平），就会发生时钟伸裁。如果允许 SCLx 引脚悬空为高电平，波特率发生器（BRG）将暂停计数直到实际采样到 SCLx 引脚为高电平为止，然后波特率发生器将被重新装入 SSPxADD<6:0> 的值并开始计数。这可以保证当外部器件将时钟拉低时，SCLx 在至少一个 BRG 周期内保持高电平 （图 19-20）。

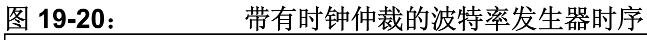
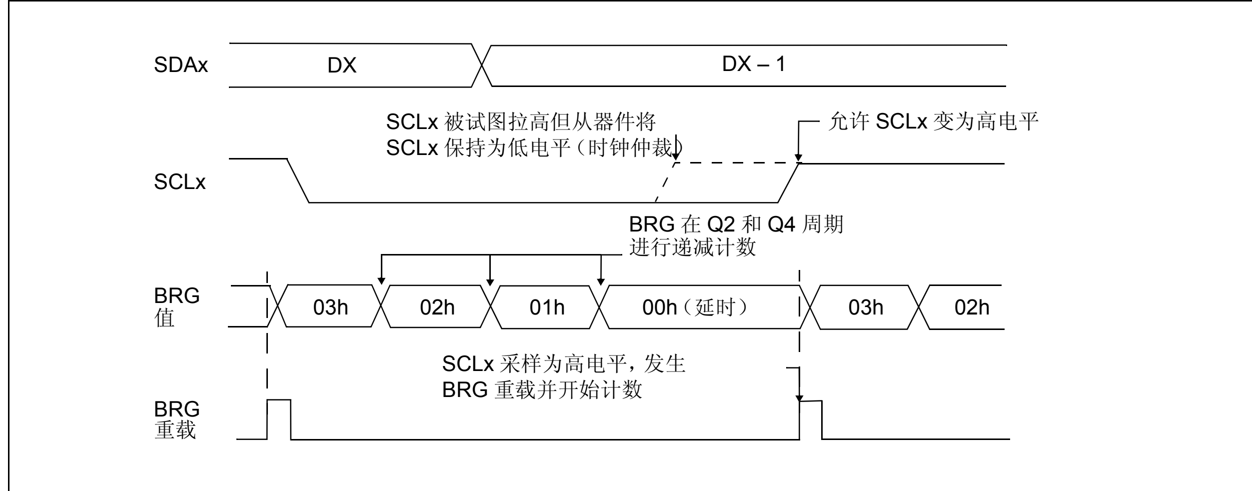


图 19-20：　带有时钟伸裁的波特率发生器时序



19.4.8 I²C 主控模式启动条件时序

要产生启动条件，用户应将启动条件使能位 SEN（SSPxCON2<0>）置 1。当 SDAx 和 SCLx 引脚采样为高电平时，波特率发生器重新装入 SSPxADD<6:0> 的内容并开始计数。如果波特率发生器发生超时（TBRG）时，SCLx 和 SDAx 都采样为高电平时，则 SDAx 引脚被驱动为低电平。当 SCLx 为高电平时，将 SDAx 驱动为低电平将产生启动条件，并使 S 位（SSPxSTAT<3>）置 1。随后波特率发生器重新装入 SSPxADD<6:0> 的值并恢复计数。当波特率发生器再次超时（TBRG）时，SEN 位（SSPxCON2<0>）将自动被硬件清零，波特率发生器暂停工作，SDAx 保持低电平，启动条件结束。

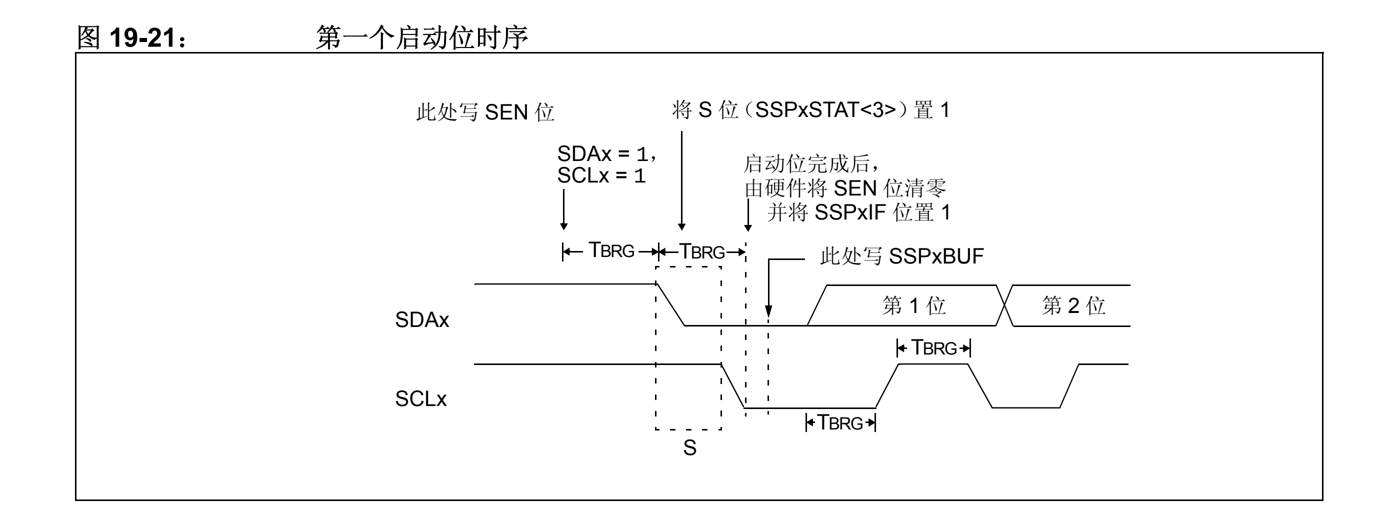
注：	如果在启动条件开始时，SDAx 和 SCLx 引脚已经采样为低电平，或者在启动条件期间，SCLx 在 SDAx 线被驱动为低电平之前已经采样为低电平，则会发生总线冲突。总线冲突中断标志位 BCLxIF 置 1，启动条件中止，I²C 模块复位到空闲状态。

19.4.8.1 WCOL 状态标志

在启动时序进行当中，如果用户写SSPxBUF，则WCOL被置 1，同时缓冲区内容不变（写操作无效）。

注：	由于不允许事件排队，在启动条件结束之前，不能写 SSPxCON2 的低 5 位。

图 19-21： 第一个启动位时序



 超前信息

19.4.9 I^2C 主控模式重复启动条件时序

将 RSEN 位（SSPxCON2<1>）编程为高电平，并且 I^2C 逻辑模块处于空闲状态时，就会产生重复启动条件。当 RSEN 位置 1 时，SCLx 引脚被拉为低电平。当 SCLx 引脚采样为低电平时，波特率发生器装入 SSPxADD<6:0> 的值，并开始计数。在该波特率发生器计数周期（TBRG）内 SDAx 引脚被释放（其引脚电平被拉高）。当波特率发生器超时时，如果 SDAx 采样为高电平， SCLx 引脚将被拉高。当 SCLx 被采样为高电平时，波特率发生器重新装入 SSPxADD<6:0> 的值并开始计数。SDAx 和 SCLx 必须在一个计数周期 TBRG 内采样为高电平。接下来，在一个 TBRG 中，将 SDAx 引脚驱动为低电平（SDAx = 0），同时 SCLx 保持高电平。然后 RSEN 位（SSPxCON2<1>）将自动清零，这次波特率发生器不会重载， SDAx 引脚保持低电平。一旦在 SDAx 和 SCLx 引脚上检测到启动条件， S 位（SSPxSTAT<3>）将被置 1。直到波特率发生器发生超时后， SSPxIF 位才会置 1。

> 注 1：有其他事件在进行时，编程设置对 RSEN 无效。
>
> 2：在重复启动条件期间，下列事件将会导致总线冲突：
> - 当 SCLx 由低电平变为高电平时，SDAx 采样为低电平。
> - 在 SDAx 被拉低之前，SCLx 变为低电平。这表明另一个主器件正试图发送一个数据 1。

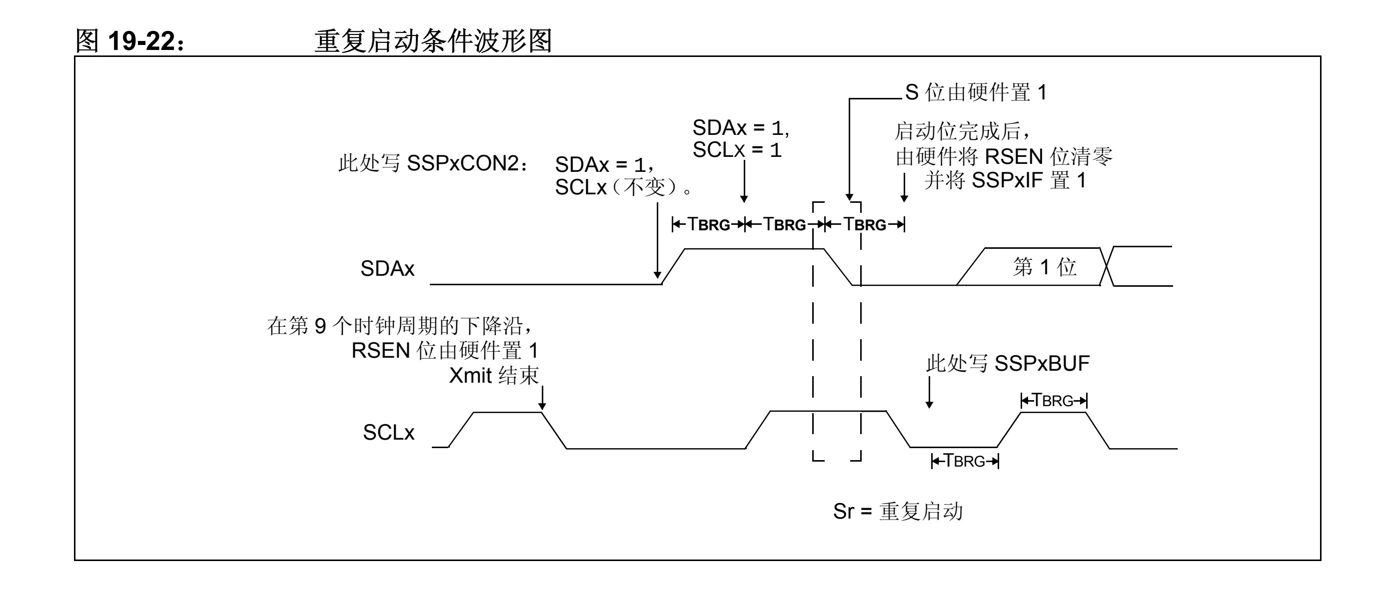
一旦 SSPxIF 位被置 1，用户便可以在 7 位地址模式下将 7 位地址，或者在 10 位地址模式下将默认的第一个地址字节写入 SSPxBUF。当发送完第一个 8 位数据并接收到一个 ACK 后，用户可以发送另外 8 位地址（10 位地址模式）或 8 位数据（7 位地址模式）。

19.4.9.1 WCOL 状态标志

在重复启动时序进行当中，如果用户写 SSPxBUF，则 WCOL 被置 1，同时缓冲区内容不变（写操作无效）。

> 注：由于不允许事件排队，在重复启动条件结束之前，不能写 SSPxCON2 的低 5 位。

图 19-22： 重复启动条件波形图



19.4.10 I^2C 主控模式下的发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的一半，都是通过写一个值到 SSPxBUF 寄存器来实现的。该操作将使缓冲区满标志位 BF 置 1，波特率发生器开始计数，同时开始下一次发送。在 SCLx 的下降沿有效后（见数据保持时间规范参数 106），地址 / 数据的每一位被移出至 SDAx 引脚。在一个波特率发生器计满返回周期（TBRG）内，SCLx 保持低电平。数据应该在 SCLx 释放为高电平前保持有效（见数据建立时间规范参数 107）。当 SCLx 引脚释放为高电平时，它将在一个 TBRG 内保持高电平状态。在此期间以及 SCLx 的下一个下降沿之后的一段时间内，SDAx 引脚上的数据必须保持稳定。在第 8 位数据被移出（第 8 个时钟周期的下降沿）之后，BF 标志位被清零，同时主器件释放 SDAx。此时如果发生地址匹配或是数据被正确接收，被寻址的从器件将在第 9 个时钟周期发出一个 ACK 位作为响应。ACK 的状态在第 9 个时钟周期的下降沿写入 ACKDT 位。主器件接收到应答之后，应答状态位ACKSTAT 会被清零。如果未收到应答，则该位被置 1。第 9 个时钟周期之后， SSPxIF 位会置 1，主控时钟 （波特率发生器）暂停，直到下一个数据字节装入 SSPxBUF， SCLx 引脚保持低电平， SDAx 保持不变 （图 19-23）。

在写 SSPxBUF 之后，地址的每一位在 SCLx 的下降沿被移出，直至所有 7 位地址位和 R/W 位都被移出。在第 8 个时钟的下降沿，主器件将释放 SDAx 引脚，以允许从器件发出一个应答响应。在第 9 个时钟的下降沿，主器件通过采样 SDAx 引脚来判断地址是否被从器件识别。ACK 位的状态被装入 ACKSTAT 状态位（SSPxCON2<6>）。在发送地址的第 9 个时钟下降沿之后，SSPxIF 置 1，BF 标志位清零，波特率发生器关闭直到下一次写 SSPxBUF，且 SCLx 引脚保持低电平，允许 SDAx 引脚悬空。

19.4.10.1 BF 状态标志

在发送模式下， BF 位 （SSPxSTAT<0>）在 CPU 写 SSPxBUF 时置 1，在所有 8 位数据移出后清零。

19.4.10.2 WCOL 状态标志

如果用户在发送进行中写 SSPxBUF（即，SSPxSR 仍在移出数据字节），则 WCOL 标志位被置 1 并且在写 SSPxBUF 之后的 2 TCY 内缓冲区的内容不变 （不发生写操作）。如果在 2 TCY 内 SSPxBUF 被重新写入，则 WCOL 位被置 1 并且 SSPBUF 被更新。这可能导致传输被破坏。

用户应在每次写 SSPxBUF 后检查 WCOL 标志位是否清零，以确保传送正确。在所有情况下， WCOL 都必须用软件清零。

19.4.10.3 ACKSTAT 状态标志

在发送模式下，当从器件发送应答响应（ACK = 0）时，ACKSTAT 位 （SSPxCON2<6>）清零；当从器件没有应答 （ACK = 1）时，该位置 1。从器件在识别出其地址（包括广播呼叫地址）或正确接收数据后，会发出一个应答。

19.4.11 I^2C 主控模式接收

通过编程接收使能位 RCEN（SSPxCON2<3>）使能主控模式接收。

注：	RCEN 位置 1 前， MSSP 必须处于空闲状态，否则上述操作无效。

波特率发生器开始计数，每次计满返回时， SCLx 引脚的状态发生改变 （由高变低或由低变高），数据被移入 SSPxSR。第 8 个时钟的下降沿之后，接收使能标志位自动清零，SSPxSR 的内容装入 SSPxBUF，BF 标志位置 1，SSPxIF 标志位置 1，波特率发生器暂停计数，且 SCLx 保持为低电平。此时 MSSP 处于空闲状态，等待下一条命令。当 CPU 读缓冲区时，BF 标志位将自动清零。通过将应答时序使能位 ACKEN（SSPxCON2<4>）置 1，用户可以在接收结束后发送应答位。

19.4.11.1 BF 状态标志

接收数据过程中，把地址或数据字节从 SSPxSR 装入 SSPxBUF 时，BF 位置 1。在读 SSPxBUF 寄存器时将其清零。

19.4.11.2 SSPOV 状态标志

接收数据过程中，当 SSPxSR 接收到 8 位数据时，SSPOV 位置 1， BF 标志位已经在上一次接收时置 1。

19.4.11.3 WCOL 状态标志

如果用户在接收过程中（即，SSPxSR 仍在移入数据字节时）写 SSPxBUF，则 WCOL 位置 1，缓冲区内容不变 （写操作无效）。

图 19-23： I²C™ 主控模式发送波形图 （7 位或 10 位地址）

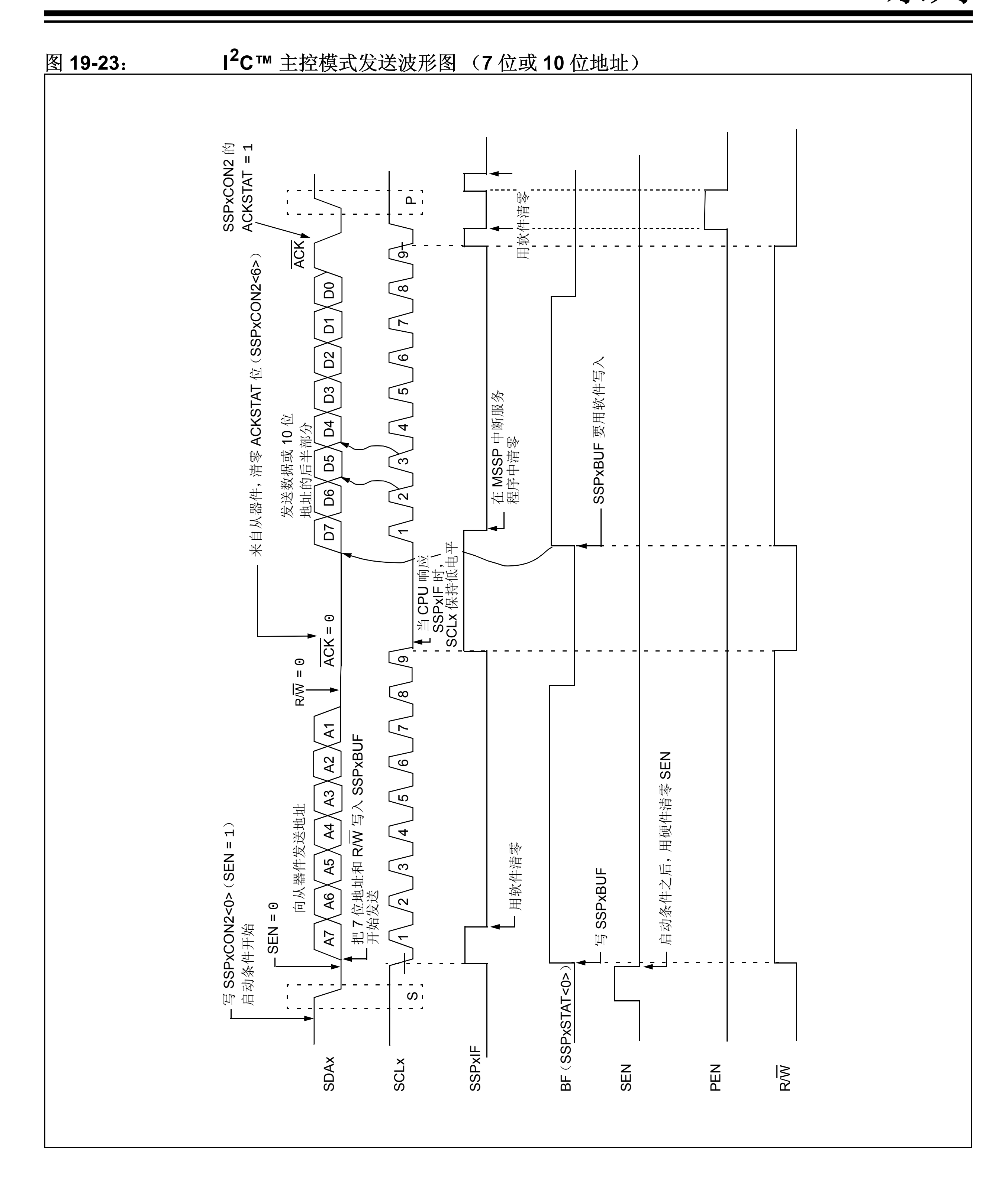
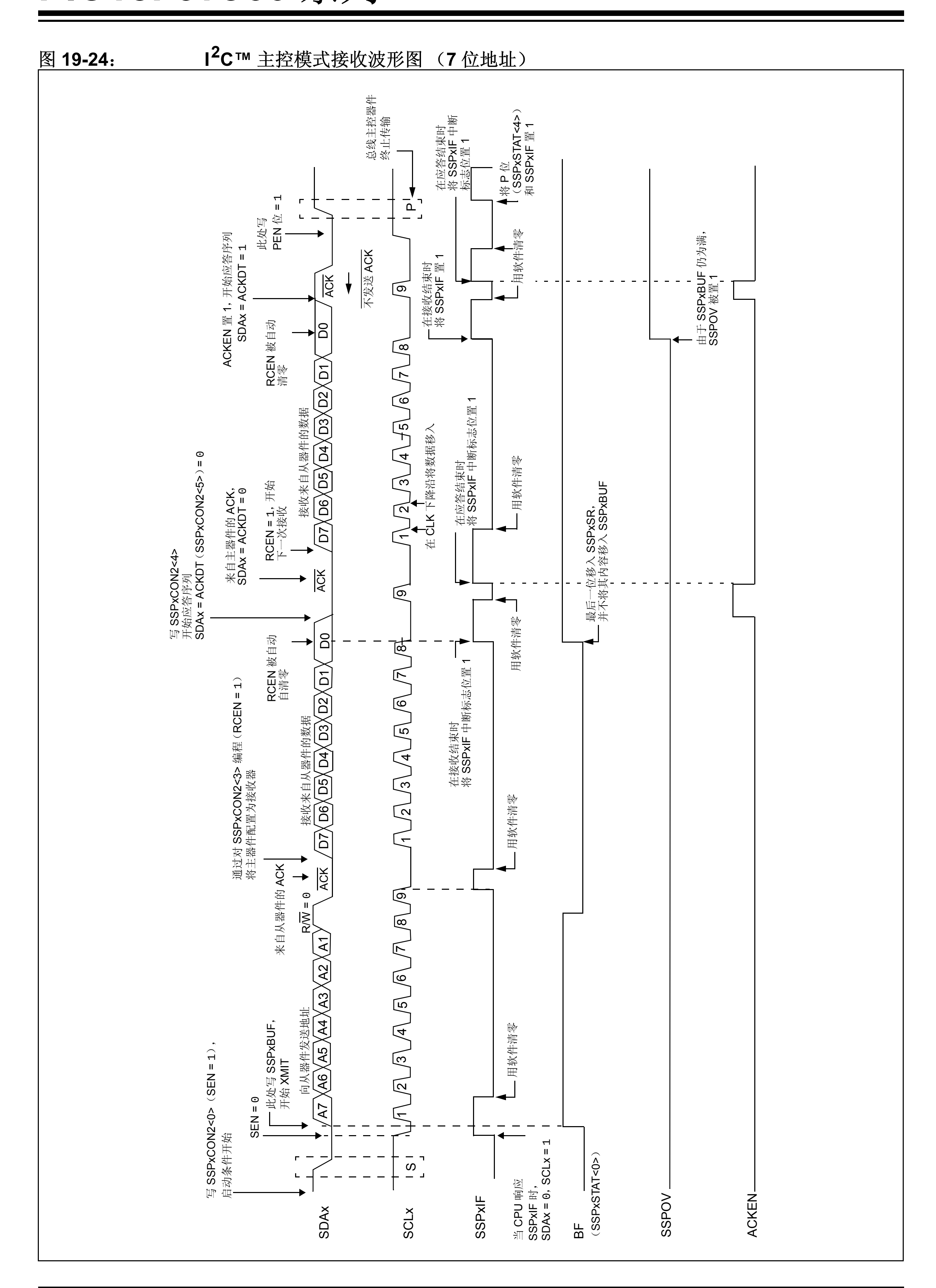


图 19-24： I²C™ 主控模式接收波形图 （7 位地址）



 超前信息

19.4.12 应答序列时序

将应答序列使能位 ACKEN （SSPxCON2<4>）置 1 即可使能应答序列。当该位被置 1 时，SCLx 引脚被拉低，应答数据位的内容输出到 SDAx 引脚上。如果用户希望产生一个应答，则应该将 ACKDT 位清零。否则，用户要在应答序列开始前将 ACKDT 位置 1。然后波特率发生器进行一个周期（TBRG）的计数，随后 SCLx 引脚电平被拉高。当 SCLx 引脚采样为高电平时 （时钟仲裁），波特率发生器再进行一个 TBRG 周期的计数。然后 SCLx 引脚被拉低。在这之后，ACKEN 位自动清零，波特率发生器关闭， MSSP 模块进入空闲模式 （图 19-25）。

19.4.12.1 WCOL 状态标志

如果用户在应答序列进行过程中试图写 SSPxBUF，则 WCOL 将置 1，缓冲区的内容不会改变（写操作无效）。

19.4.13 停止条件时序

如果将停止时序使能位 PEN （SSPxCON2<2>）置 1，则在接收 / 发送结束后，SDAx 引脚上将产生停止位。在接收 / 发送结束时，SCLx 引脚在第 9 个时钟的下降沿后保持低电平。当 PEN 位置 1 时，主器件将 SDAx 线置为低电平。当 SDAx 线采样为低电平时，波特率发生器被重载并递减计数至 0。当波特率发生器发生超时，SCLx 引脚被拉为高电平，在一个 TBRG （波特率发生器计满返回周期）之后，SDAx 引脚将被拉高。当 SDAx 引脚采样为高电平且 SCLx 也是高电平时， P 位（SSPxSTAT<4>）置 1。另一个 TBRG 之后，PEN 位被清零，同时 SSPxIF 位被置 1 （图 19-26）。

19.4.13.1 WCOL 状态标志

如果用户在停止时序过程中试图写 SSPxBUF，则 WCOL 位将置 1，缓冲区的内容不会改变 （写操作无效）。

图 19-25： 应答序列波形图

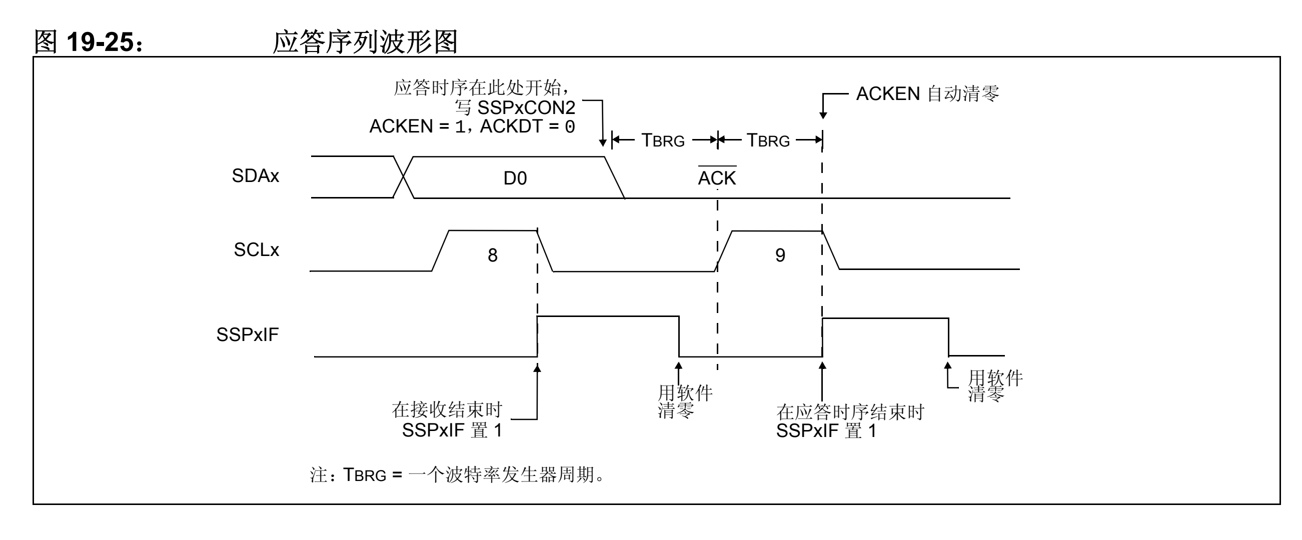
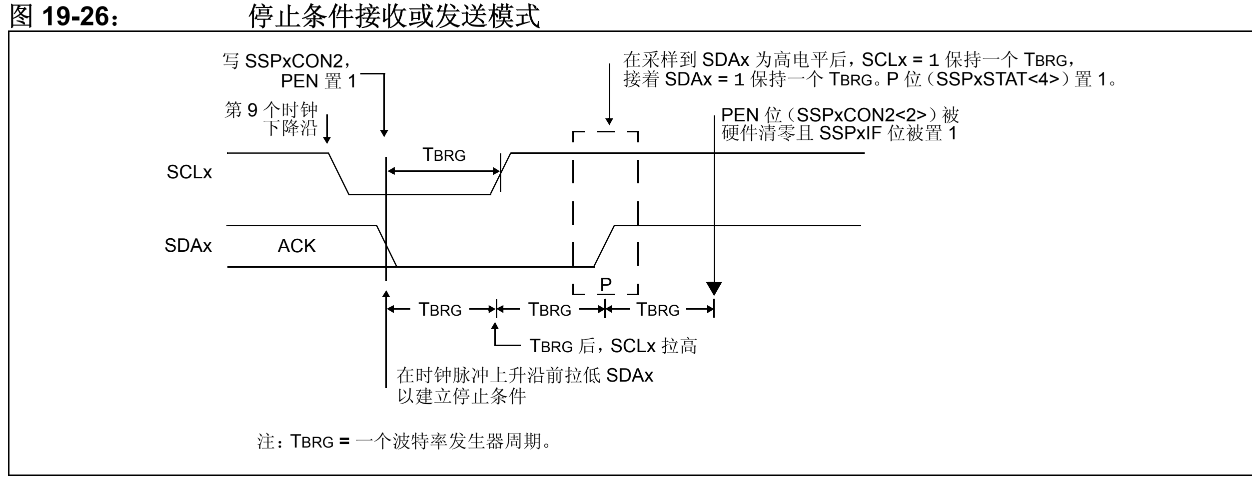


图 19-26： 停止条件接收或发送模式



19.4.14 休眠模式下的操作

在休眠模式下，I^2C 模块能够接收地址或数据，并且在地址匹配或字节传输完成后，如果允许 MSSP 中断，将唤醒处理器。

19.4.15 复位的影响

复位操作会禁止 MSSP 模块并终止当前的数据传输。

19.4.16 多主机模式

在多主机模式下，在检测到启动和停止条件时将产生中断，这可以用于判断总线是否空闲。停止（P）位和启动（S）位在复位或禁止 MSSP 模块时清零。当 P 位（SSPxSTAT<4>）置 1 时，可以取得 I^2C 总线的控制权；否则，S 位和 P 位都清零，总线处于空闲状态。当总线忙时，一旦出现停止条件，将产生 MSSP 中断。

在多主机模式下，必须一直监视 SDAx 线，查看信号电平是否为期望的输出电平。此操作由硬件实现，其结果保存在 BCLxIF 位中。

可能导致仲裁失败的情况是：

- 地址传输
- 数据传输
- 启动条件
- 重复的启动条件
- 应答条件

19.4.17 多主机通信、总线冲突与总线仲裁

多主机模式是通过总线仲裁来支持的。当主器件将地址 / 数据位输出到 SDAx 引脚时，如果一个主器件在 SDAx 上输出 1（将 SDAx 引脚悬空为高电平），而另一个主器件输出 0，就会发生总线仲裁。当 SCLx 引脚悬空为高电平时，数据必须是稳定的。如果 SDAx 引脚上期望的数据是 1，而实际采样到的数据是 0，则发生了总线冲突。主器件将把总线冲突中断标志位 BCLxIF 置 1，并将 I^2C 端口复位到空闲状态（图 19-27）。

如果在发送过程中发生总线冲突，则发送操作停止，BF 标志位被清零，SDAx 和 SCLx 线被拉高，并且将 SSPxBUF置于可写入状态。当执行完总线冲突中断服务程序后，如果 I^2C 总线空闲，用户可通过发出启动条件恢复通信。

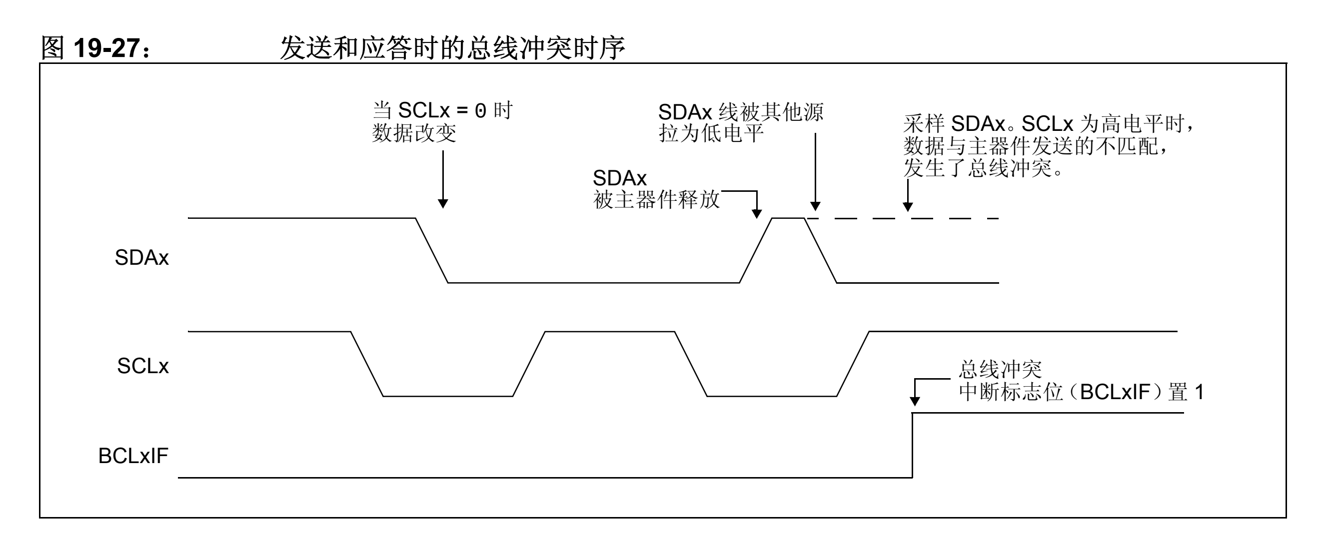
如果在启动、重复启动、停止或应答信号的执行过程中发生总线冲突，则这种状态被中止，SDAx 和 SCLx 线被拉高，SSPxCON2 寄存器中的对应控制位清零。当执行完总线冲突中断服务程序后，如果 I^2C 总线空闲，用户可通过发出启动条件恢复通信。

主器件将继续监视 SDAx 和 SCLx 引脚。一旦出现停止条件，SSPxIF 位将被置 1。

发生总线冲突时无论发送的进度如何，写入 SSPxBUF 都会从第一个数据位开始发送数据。

在多主机模式下，通过在检测到启动和停止条件时产生中断可以确定总线何时空闲。SSPxSTAT 寄存器中的 P 位置 1 时，可以获取 I^2C 总线的控制权；否则，S 位和 P 位都清零，总线处于空闲状态。

图 19-27： 发送和应答时的总线冲突时序



 超前信息

19.4.17.1 启动条件期间的总线冲突

启动条件期间，以下事件将导致总线冲突：

a） 在启动条件开始时， SDAx 或 SCLx 被采样为低电平 （图 19-28）。

b） SDAx 被拉低之前，SCLx 采样为低电平（图 19-29）。

在启动条件期间， SDAx 和 SCLx 引脚都会被监视。

如果 SDAx 引脚已经是低电平，或 SCLx 引脚已经是低电平，则：

- 中止启动条件，
- BCLxIF 标志位置 1，
- MSSP 模块复位为空闲状态 （图 19-28）。

启动条件从 SDAx 和 SCLx 引脚被拉高开始。当 SDAx 引脚采样为高电平时，波特率发生器装入 SSPxADD<6:0> 的值并递减计数至 0。如果在 SDAx 为高电平时， SCLx 引脚采样为低电平，则发生了总线冲突，因为这表示另一个主器件在启动条件期间试图发送一个数据 1。

如果 SDAx 引脚在该计数周期内采样为低电平，则 BRG 复位，同时 SDAx 线保持原值 （图 19-30）。但是，如果 SDAx 引脚采样为 1，则在 BRG 计数结束时该引脚将被置为低电平。接着，波特率发生器被重载并递减计数至 0，在此期间，如果 SCLx 引脚采样到 0，则不会发生总线冲突。在 BRG 计数结束时，SCLx 引脚被拉为低电平。

注：	在启动条件期间不太可能发生总线冲突，因为两个总线主控器件不可能精确地在同一时刻发出启动条件。因此一个主器件将总是先于另一个主器件将 SDAx 拉低。但是上述情况不会引起总线冲突，因为两个主器件一定会对启动条件后的第一个地址进行仲裁。如果地址是相同的，必须继续对数据部分、重复启动条件或停止条件进行仲裁。

图 19-28： 启动条件期间的总线冲突 （仅 **SDAx**）

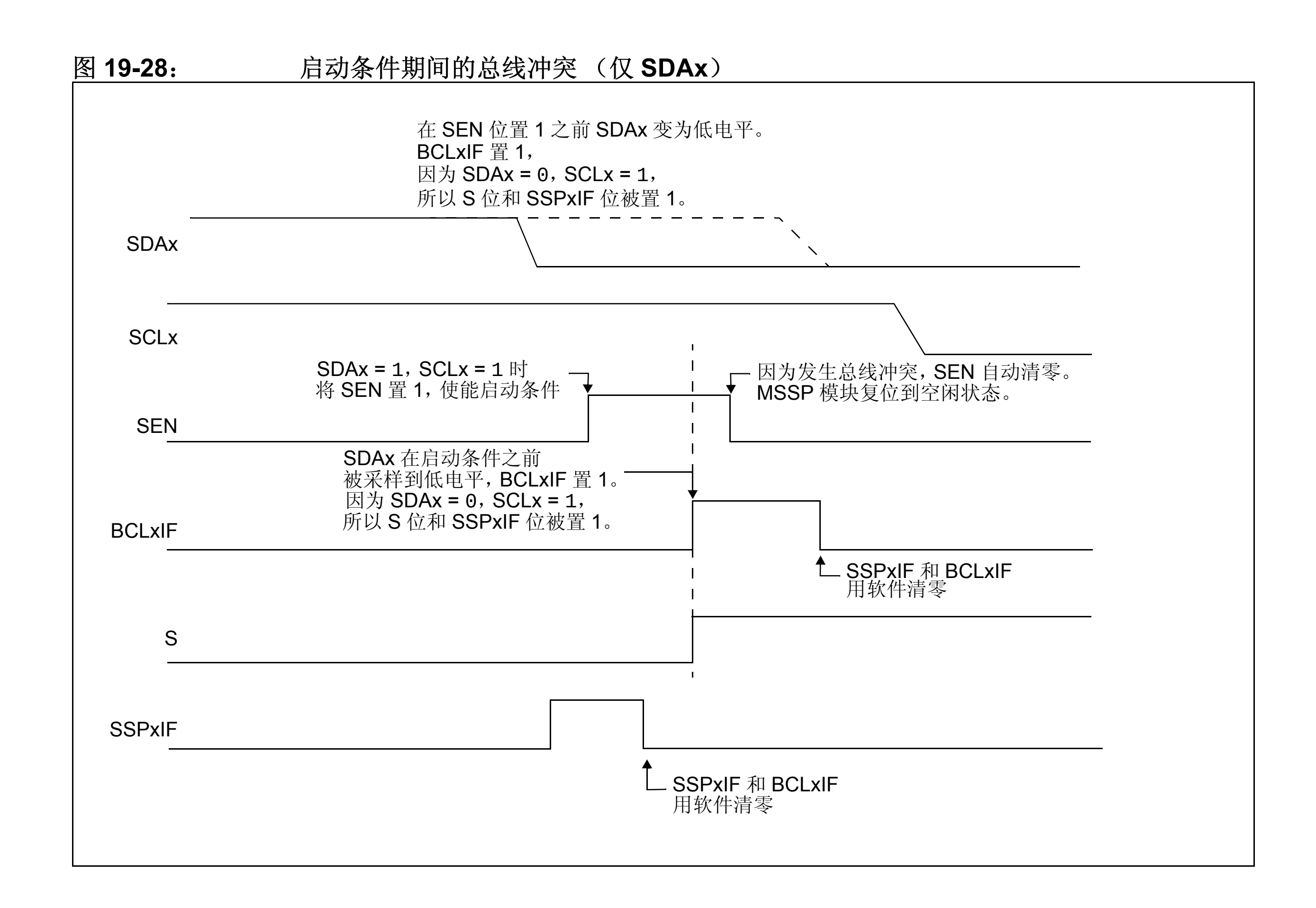


图 19-29： 启动条件期间的总线冲突 （**SCLx = 0**）

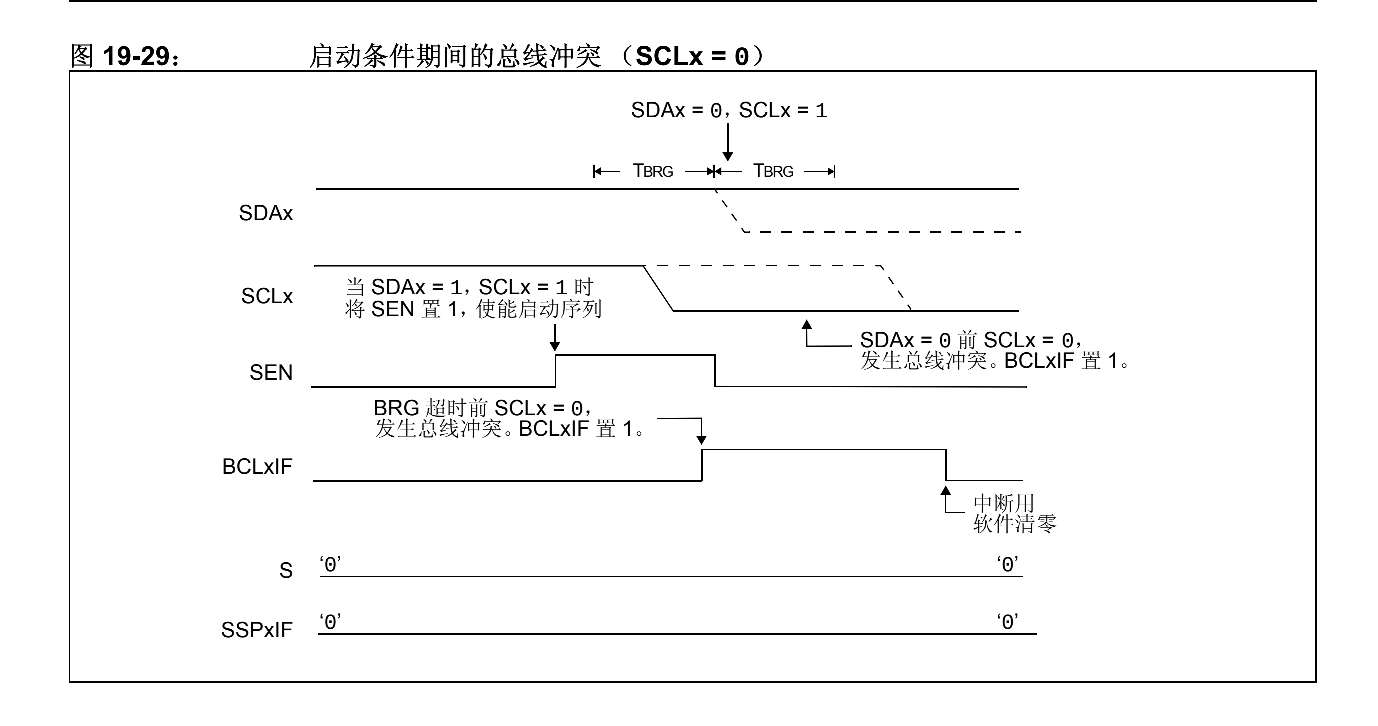
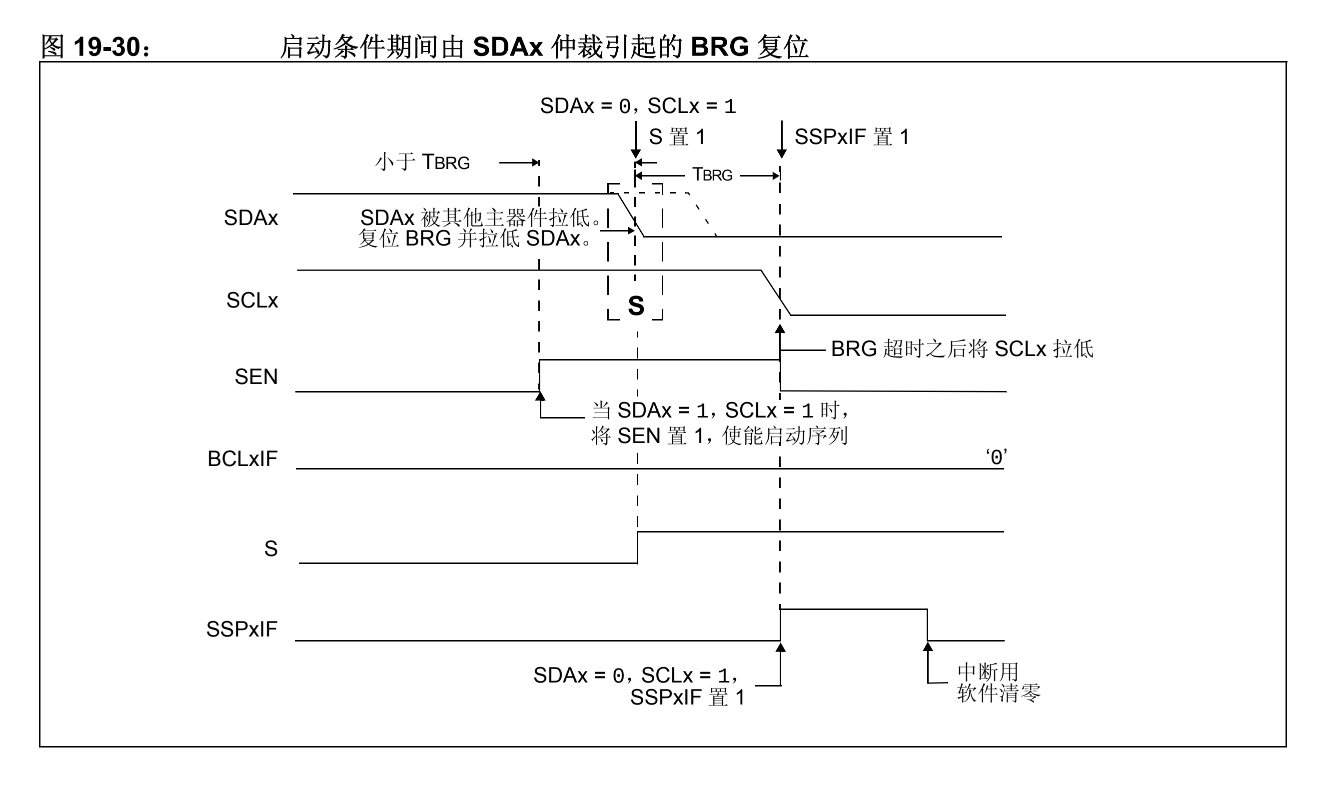


图 19-30： 启动条件期间由 **SDAx** 仲裁引起的 **BRG** 复位



 超前信息

19.4.17.2 重复启动条件期间的总线冲突

在下列情况中，重复启动条件期间会发生总线冲突：

a） 在 SCLx 由低电平变为高电平期间，在 SDAx 上采样到低电平。

b） 在 SDAx 被拉为低电平之前，SCLx 变为低电平，表示另一个主器件正试图发送一个数据 1。

当用户释放 SDAx 并允许该引脚悬空时， BRG 装入 SSPxADD<6:0> 中的值并递减计数至 0，接着 SCLx 引脚被拉高，当 SCLx 引脚采样到高电平时，对 SDAx 引脚进行采样。

如果 SDAx 为低电平，则已发生了总线冲突（即，另一个主器件正试图发送一个数据 0，见图 19-31）。如果 SDAx 被采样到高电平，则 BRG 被重新装入值并开始计数。如果 SDAx 在 BRG 超时之前从高电平变为低电平，则不会发生总线冲突，因为两个主器件不可能精确地在同一时刻将 SDAx 拉低。

如果 SCLx 在 BRG 超时之前从高电平变为低电平，且 SDAx 尚未被拉低，那么将发生总线冲突。在此情况下，另一个主器件在重复启动条件期间正试图发送一个数据 1（见图 19-32）。

如果在 BRG 计时结束时 SCLx 和 SDAx 都仍然是高电平，则 SDAx 引脚被拉低， BRG 重新装入值并开始计数。在计数结束时，不管 SCLx 引脚的状态如何，SCLx 引脚都被拉低，重复启动条件结束。

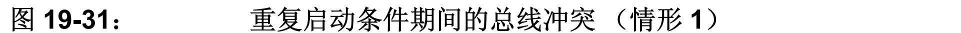


图 19-31： 重复启动条件期间的总线冲突（情形 1）

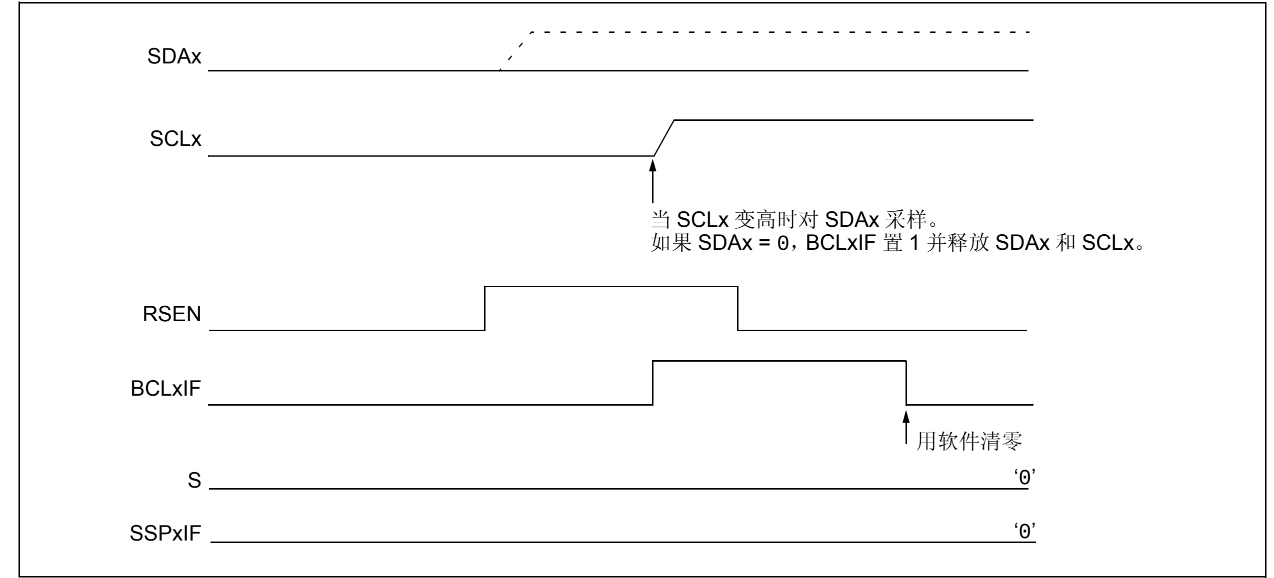
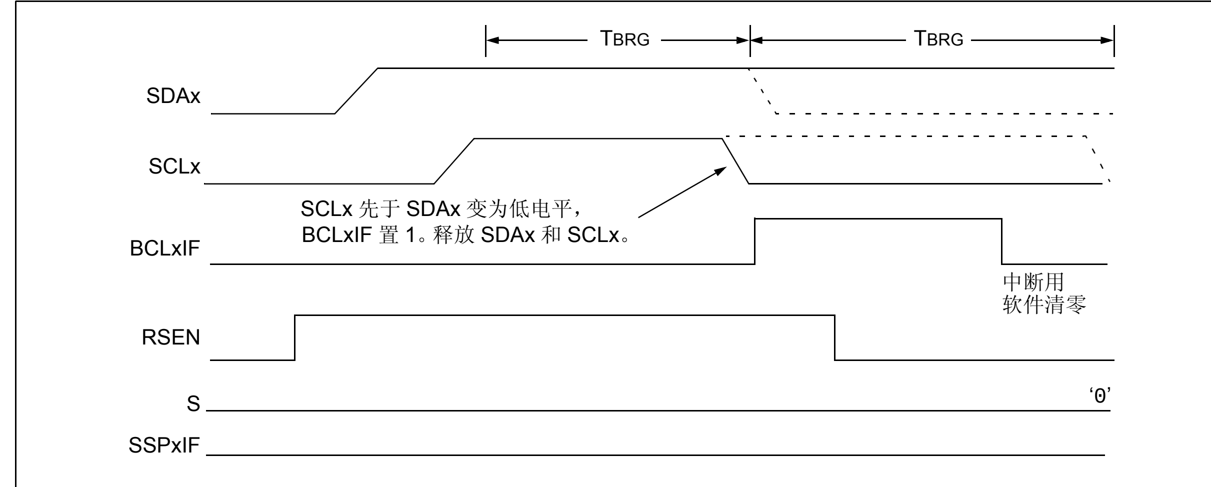


图 19-32： 重复启动条件期间的总线冲突（情形 2）



19.4.17.3 停止条件期间的总线冲突

以下事件会导致停止条件期间发生总线冲突：

a） SDAx 已被拉高并允许悬空为高电平之后，SDAx 在 BRG 超时后被采样到低电平。

b） SCLx 引脚被拉高之后，SCLx 在 SDAx 变成高电平之前被采样到低电平。

停止条件从 SDAx 被置成低电平开始。当 SDAx 采样为低电平时，SCLx 引脚被允许悬空。当 SDAx 被采样到高电平时（时钟仲裁），波特率发生器装入SSPxADD<6:0>的值并递减计数至 0。 BRG 超时后， SDAx 被采样。如果 SDAx 采样为低电平，则已发生总线冲突。这是因为另一个主器件正试图发送一个数据 0（图 19-33）。如果 SCLx 引脚在允许 SDAx 悬空为高电平前被采样到低电平，也会发生总线冲突。这是另一个主器件正试图发送一个数据 0 的另外一种情况 （图 19-34）。

图 19-33： 停止条件期间的总线冲突 （情形 1）

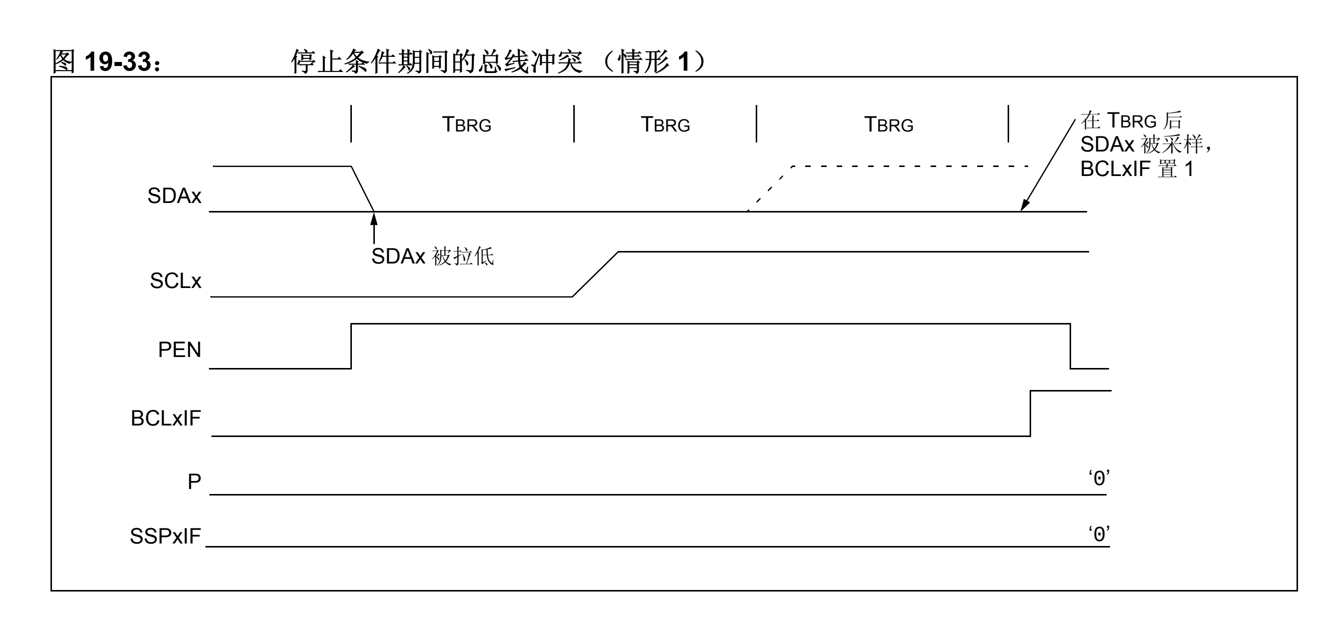
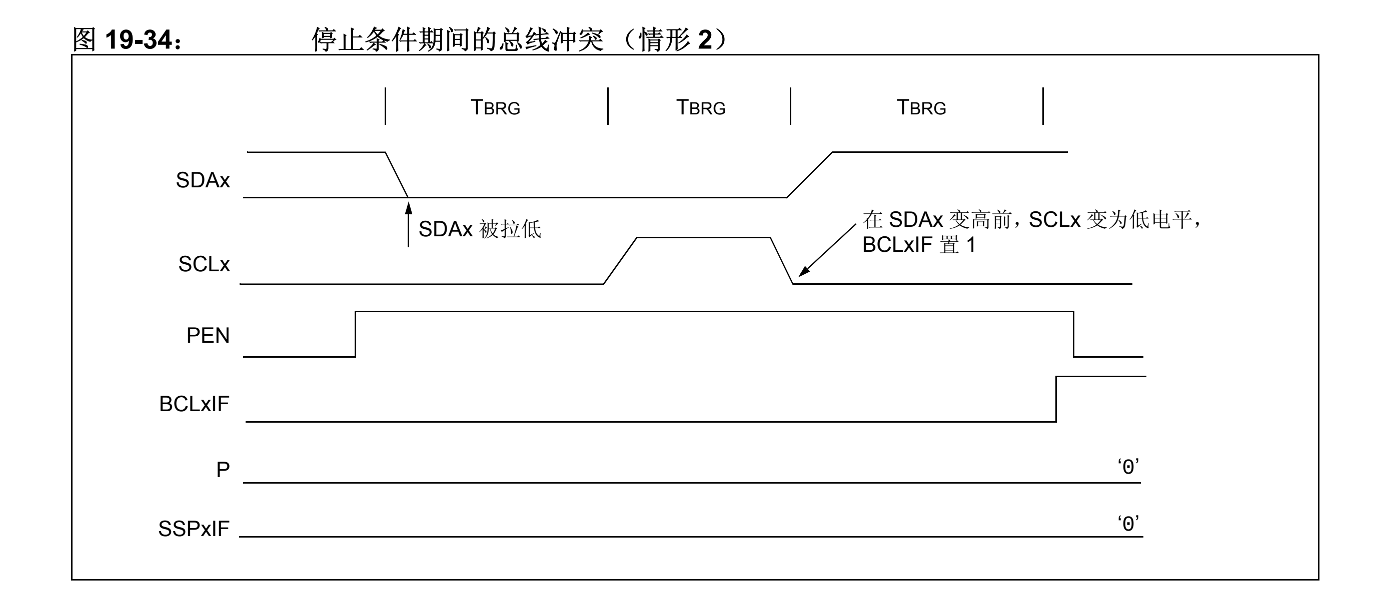


图 19-34： 停止条件期间的总线冲突 （情形 2）



 超前信息

表 19-4： 与 I^{2}C™ 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	61
PIR3	SSP2IF(1)	BCL2IF(1)	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE(1)	BCL2IE(1)	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP(1)	BCL2IP(1)	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	61
TRISD	TRISD7	TRISD6(1)	TRISD5(1)	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	61
SSP1BUF	MSSP1 接收缓冲区 / 发送寄存器								60
SSP1ADD	MSSP1 地址寄存器 （I^{2}C™ 从动模式）， MSSP1 波特率重载寄存器 （I^{2}C 主控模式）								63
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	60
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN	
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	60
SSP2BUF	MSSP2 接收缓冲区 / 发送寄存器								60
SSP2ADD	MSSP2 地址寄存器 （I^{2}C 从动模式）， MSSP2 波特率重载寄存器 （I^{2}C 主控模式）								63
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	63
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	63
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN	
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	63

图注： — = 未用 （读为 0）， r = 保留。 I^{2}C™ 模式下的 MSSP 模块不使用阴影单元。

注 1： 这些位仅在 100 引脚器件上实现；在其他器件上未实现且读为 0。

2： I^{2}C™ 从动模式下的位定义。

注：

20.0 增强型通用同步 / 异步收发器（EUSART）

增强型通用同步 / 异步收发器（Enhanced Universal Synchronous Asynchronous Receiver Transmitter，EUSART）模块是两个串行 I/O 模块之一。（EUSART 也称为“串行通信接口”或 SCI。）可以将 EUSART 配置为能与 CRT 终端和个人计算机等外设通信的全双工异步系统，也可以将它配置成能够与 A/D 或 D/A 集成电路、串行 EEPROM 等外设通信的半双工同步系统。

增强型 USART 模块还实现了其他功能，包括自动波特率检测和校准、接收到同步间隔字符时的自动唤醒和 12 位间隔字符发送。因为具有这些功能，所以用在局域互连网络（Local Interconnect Network，LIN）总线系统中非常理想。

PIC18F97J60 系列的 64 引脚器件配备有一个 EUSART 模块，称为 EUSART1。80 引脚和 100 引脚器件都有两个独立的 EUSART 模块，分别称为 EUSART1 和 EUSART2。可将这两个模块配置为以下几种工作模式：

- 带有以下功能的异步模式（全双工）：
  - 接收到字符时自动唤醒
  - 自动波特率校准
  - 12 位间隔字符发送
- 同步——主控（半双工）模式，时钟极性可选
- 同步——从动（半双工）模式，时钟极性可选

EUSART1 和 EUSART2 的引脚分别与 PORTC（RC6/TX1/CK1 和 RC7/RX1/DT1）和 PORTG（RG1/TX2/CK2 和 RG2/RX2/DT2）的功能复用。要把这些引脚配置为 EUSART：

- 对于 EUSART1：
  - SPEN（RCSTA1<7>）位必须置 1（= 1）
  - TRISC<7> 位必须置 1（= 1）
  - TRISC<6> 位必须清零（= 0）使该模块工作于异步和同步主控模式
  - TRISC<6> 位必须置 1（= 1）使该模块工作于同步从动模式
- 对于 EUSART2：
  - SPEN（RCSTA2<7>）位必须置 1（= 1）
  - TRISG<2> 位必须置 1（= 1）
  - TRISG<1> 位必须清零（= 0）使该模块工作于异步和同步主控模式
  - TRISC<6> 位必须置 1（= 1）使该模块工作于同步从动模式

注：	EUSARTx 控制逻辑在需要时会自动将引脚从输入重新配置为输出。

增强型 USART 模块的操作是由以下 3 个寄存器控制的：

- 发送状态和控制寄存器（TXSTAx）
- 接收状态和控制寄存器（RCSTAx）
- 波特率控制寄存器（BAUDCONx）

这些寄存器将在寄存器 20-1、寄存器 20-2 和寄存器 20-3 中分别详细介绍。

注：	在本节中，凡是涉及与特定 EUSART 模块相关的寄存器和位的名称一般都采用以“x”代替特定模块编号的方式。因此，“RCSTAx”可能指 EUSART1 的接收状态寄存器，也可能指 EUSART2 的接收状态寄存器。

寄存器 20-1： TXSTAx：发送状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN[1]	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **CSRC**：时钟源选择位
<u>异步模式：</u>
无关位。
<u>同步模式：</u>
1 = 主控模式（时钟来自内部 BRG）
0 = 从动模式（时钟来自外部时钟源）

bit 6 **TX9**：9 位发送使能位
1 = 选择 9 位发送
0 = 选择 8 位发送

bit 5 **TXEN**：发送使能位 [1]
1 = 使能发送
0 = 禁止发送

bit 4 **SYNC**：EUSARTx 模式选择位
1 = 同步模式
0 = 异步模式

bit 3 **SENDB**：发送间隔字符位
<u>异步模式：</u>
1 = 在下一次发送时发送 "同步间隔" 字符（在完成时由硬件清零）
0 = "同步间隔" 字符发送完成
<u>同步模式：</u>
无关位。

bit 2 **BRGH**：高波特率选择位
<u>异步模式：</u>
1 = 高速
0 = 低速
<u>同步模式：</u>
在此模式下未使用。

bit 1 **TRMT**：发送移位寄存器状态位
1 = TSR 空
0 = TSR 满

bit 0 **TX9D**：发送数据的第 9 位
可以是地址 / 数据位或奇偶校验位。

注 1：在同步模式下， SREN/CREN 的优先级高于 TXEN。

寄存器 20-2： RCSTAx：接收状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SPEN：**串口使能位
1 = 使能串口 （配置 RXx/DTx 和 TXx/CKx 引脚作为串口引脚）
0 = 禁止串口 （保持在复位状态）

bit 6 **RX9：**9 位接收使能位
1 = 选择 9 位接收
0 = 选择 8 位接收

bit 5 **SREN：**单字节接收使能位
异步模式：
无关位。
同步主控模式：
1 = 使能单字节接收
0 = 禁止单字节接收
此位在接收完成后清零。
同步从动模式：
无关位。

bit 4 **CREN：**连续接收使能位
异步模式：
1 = 使能接收器
0 = 禁止接收器
同步模式：
1 = 使能连续接收，直到使能位 CREN 清零 （CREN 比 SREN 优先级高）
0 = 禁止连续接收

bit 3 **ADDEN：**地址检测使能位
9 位异步模式 （RX9 = 1）：
1 = 当 RSR<8> 置 1 时，使能地址检测、允许中断和装入接收缓冲区
0 = 禁止地址检测、接收所有字节并且第 9 位可作为奇偶校验位
9 位异步模式 （RX9 = 0）：
无关位。

bit 2 **FERR：**帧错误位
1 = 帧错误 （可以通过读 RCREGx 寄存器刷新该位并接收下一个有效字节）
0 = 无帧错误

bit 1 **OERR：**溢出错误位
1 = 溢出错误 （可以通过清除 CREN 位清零）
0 = 无溢出错误

bit 0 **RX9D：**接收数据的第 9 位
该位可以是地址 / 数据位或奇偶校验位，并且必须由用户固件计算得到。

寄存器 20-3： BAUDCONx：波特率控制寄存器

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **ABDOVF：**自动波特率采样进位状态位
1 = 在自动波特率检测模式下出现了 BRG 进位 （必须用软件清零）
0 = 没有发生 BRG 进位

bit 6 **RCIDL：**接收操作空闲状态位
1 = 接收操作处于空闲状态
0 = 接收操作处于活动状态

bit 5 **RXDTP：**接收数据极性选择位
异步模式：
1 = RXx 数据反相
0 = 接收的 RXx 数据未反相
同步模式：
1 = CKx 时钟反相
0 = CKx 时钟未反相

bit 4 **TXCKP：**时钟和数据极性选择位
异步模式：
1 = TXx 数据反相
0 = TXx 数据未反相
同步模式：
1 = CKx 时钟反相
0 = CKx 时钟未反相

bit 3 **BRG16：**16 位波特率寄存器使能位
1 = 16 位波特率发生器——SPBRGHx 和 SPBRGx
0 = 8 位波特率发生器——仅 SPBRGx，忽略 SPBRGHx 的值 （兼容模式）

bit 2 未用：读为 0

bit 1 **WUE：**唤醒使能位
异步模式：
1 = EUSARTx 将继续采样 RXx 引脚——中断在下降沿产生，在下一个上升沿由硬件清零该位
0 = 未监测 RXx 引脚或检测到了上升沿
同步模式：
在此模式下未使用。

bit 0 **ABDEN：**自动波特率检测使能位
异步模式：
1 = 在下一个字符使能波特率检测。需要收到 “同步”字段 （55h），完成时由硬件清零
0 = 禁止波特率检测或检测已完成
同步模式：
在此模式下未使用。

20.1 波特率发生器 （BRG）

BRG 是一个专用的 8 位或 16 位发生器，支持 EUSARTx 的异步和同步模式。默认情况下，BRG 工作在 8 位模式下，通过将 BRG16 位（BAUDCONx<3>）置 1 可以选择 16 位模式。

SPBRGHx:SPBRGx 寄存器对控制自由运行的定时器周期。在异步模式下， BRGH （TXSTAx<2>）和 BRG16 （BAUDCONx<3>）也用于控制波特率。在同步模式下，BRGH 位会被忽略。表 20-1 所示为不同 EUSARTx 模式的波特率计算公式，但仅适用于主控模式 （由内部产生时钟信号）。

给出期望的波特率和 FOSC 值，就可以使用表 20-1 中的公式计算 SPBRGHx:SPBRGx 寄存器的最近似整数值。这样就可以判断波特率误差。例 20-1 给出了计算示例。表 20-2 中给出了不同异步模式下典型的波特率和误差值。使用高波特率 （BRGH = 1）或 16 位 BRG 有利于减小波特率误差，或者在快速振荡频率条件下实现低波特率。

向 SPBRGHx:SPBRGx 寄存器写入新值会使 BRG 定时器复位 （或清零）。这可以确保 BRG 无需等待定时器溢出就可以输出新的波特率。

20.1.1 在功耗管理模式下的操作

器件时钟用于产生所需的波特率。当进入一种功耗管理模式时，新时钟源可能会工作在一个不同的频率下。这可能需要调整 SPBRGx 寄存器对中的值。

20.1.2 采样

检测电路对RXx引脚（RC7/RX1/DT1或RG2/RX2/DT2）采样三次，以判定 RXx 引脚上出现的是高电平还是低电平。

表 20-1： 波特率公式

配置位			BRG/EUSARTx 模式	波特率计算公式
SYNC	BRG16	BRGH		
0	0	0	8 位 / 异步	FOSC/[64 (n + 1)]
0	0	1	8 位 / 异步	FOSC/[16 (n + 1)]
0	1	0	16 位 / 异步	
0	1	1	16 位 / 异步	FOSC/[4 (n + 1)]
1	0	x	8 位 / 同步	
1	1	x	16 位 / 同步	

图注： x = 无关位， n = SPBRGHx:SPBRGx 寄存器对的值

例 20-1： 计算波特率误差

器件工作在 Fosc = 16 MHz，目标波特率 = 9600，异步模式， 8 位 BRG：

目标波特率 = Fosc/(64 ([SPBRGHx:SPBRGx] + 1))

求解 SPBRGHx:SPBRGx：

X = ((Fosc/ 目标波特率)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25

波特率计算结果 = 16000000/(64 (25 + 1))
= 9615

误差 = (波特率计算结果 – 目标波特率)/ 目标波特率
= (9615 – 9600)/9600 = 0.16%

表 20-2： 与波特率发生器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用，读为 0。 BRG 不使用阴影单元。

 超前信息

表 **20-3**： 异步模式下的波特率

波特率 (K)	SYNC = 0， BRG16 = 0， BRGH = 0											
	Fosc = 41.667 MHz			Fosc = 31.25 MHz			Fosc = 25.000 MHz			Fosc = 20.833 MHz		
	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	1.271	5.96	255
2.4	2.543	5.96	255	2.405	0.22	202	2.396	-0.15	162	2.393	-0.27	135
9.6	9.574	-0.27	67	9.574	-0.27	50	9.527	-0.76	40	9.574	-0.27	33
19.2	19.148	-0.27	33	19.531	1.73	24	19.531	1.73	19	19.147	-0.27	16
57.6	59.186	2.75	10	61.035	5.96	7	55.804	-3.12	6	54.253	-5.81	5
115.2	108.508	-5.81	5	122.070	5.96	3	130.208	13.03	2	108.505	-5.81	2

波特率 (K)	SYNC = 0， BRG16 = 0， BRGH = 0								
	Fosc = 13.889 MHz			Fosc = 6.250 MHz			Fosc = 4.167 MHz		
	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	0.300	0.01	216
1.2	1.198	-0.08	180	1.206	0.47	80	1.206	0.48	53
2.4	2.411	0.47	89	2.382	-0.76	40	2.411	0.48	26
9.6	9.435	-1.71	22	9.766	1.73	9	9.301	-3.11	6
19.2	19.279	2.75	10	19.531	1.73	4	21.703	13.04	2
57.6	54.254	-5.81	3	48.828	-15.23	1	65.109	13.04	0
115.2	108.508	-5.81	1	97.656	-15.23	0	65.109	-43.48	0

波特率 (K)	SYNC = 0， BRG16 = 0， BRGH = 1											
	Fosc = 41.667 MHz			Fosc = 31.25 MHz			Fosc = 25.000 MHz			Fosc = 20.833 MHz		
	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	—	—	—	—	—	—
9.6	10.172	5.96	255	9.621	0.22	202	9.586	-0.15	162	9.573	-0.27	135
19.2	19.148	-0.27	135	19.148	-0.27	101	19.290	0.47	80	19.147	-0.27	67
57.6	57.871	0.47	44	57.445	-0.27	33	57.870	0.47	26	56.611	-1.72	22
115.2	113.226	-1.71	22	114.890	-0.27	16	111.607	-3.12	13	118.369	2.75	10

波特率 (K)	SYNC = 0， BRG16 = 0， BRGH = 1								
	Fosc = 13.889 MHz			Fosc = 6.250 MHz			Fosc = 4.167 MHz		
	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）	实际波特率 (K)	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	1.200	0.01	216
2.4	—	—	—	2.396	-0.15	162	2.389	-0.44	108
9.6	9.645	0.47	89	9.527	-0.76	40	9.645	0.48	26
19.2	19.290	0.47	44	19.531	1.73	19	18.603	-3.11	13
57.6	57.871	0.47	14	55.804	-3.12	6	52.088	-9.57	4
115.2	108.508	-5.81	7	130.208	13.03	2	130.219	13.04	1

表 20-3：　　　异步模式下的波特率 （续）

波特率 (K)	SYNC = 0， BRG16 = 1， BRGH = 0											
	Fosc = 41.667 MHz			Fosc = 31.25 MHz			Fosc = 25.000 MHz			Fosc = 20.833 MHz		
	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)
0.3	0.300	0.00	8680	0.300	0.00	6509	0.300	0.01	5207	0.300	0.00	4339
1.2	1.200	0.01	2169	1.200	-0.02	1627	1.200	0.01	1301	1.200	0.00	1084
2.4	2.400	0.01	1084	2.399	-0.02	813	2.400	0.01	650	2.398	-0.09	542
9.6	9.609	0.10	270	9.621	0.22	202	9.586	-0.15	162	9.574	-0.27	135
19.2	19.148	-0.27	135	19.148	-0.27	101	19.290	0.47	80	19.148	-0.27	67
57.6	57.871	0.47	44	57.444	-0.27	33	57.870	0.47	26	56.611	-1.72	22
115.2	113.226	-1.71	22	114.890	-0.27	16	111.607	-3.12	13	118.369	2.75	10

波特率 (K)	SYNC = 0， BRG16 = 1， BRGH = 0								
	Fosc = 13.889 MHz			Fosc = 6.250 MHz			Fosc = 4.167 MHz		
	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)
0.3	0.300	-0.02	2893	0.300	0.01	1301	0.300	0.01	867
1.2	1.201	0.05	722	1.198	-0.15	325	1.200	0.01	216
2.4	2.398	-0.08	361	2.396	-0.15	162	2.389	-0.44	108
9.6	9.645	0.47	89	9.527	-0.76	40	9.646	0.48	26
19.2	19.290	0.47	44	19.531	1.73	19	18.603	-3.11	13
57.6	57.871	0.47	14	55.804	-3.12	6	52.088	-9.57	4
115.2	108.508	-5.81	7	130.208	13.03	2	130.218	13.04	1

波特率 (K)	SYNC = 0， BRG16 = 1， BRGH = 1 或 SYNC = 1， BRG16 = 1											
	Fosc = 41.667 MHz			Fosc = 31.25 MHz			Fosc = 25.000 MHz			Fosc = 20.833 MHz		
	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)
0.3	0.300	0.00	34722	0.300	0.00	26041	0.300	0.00	20832	0.300	0.00	17360
1.2	1.200	0.00	8680	1.200	0.01	6509	1.200	0.01	5207	1.200	0.00	4339
2.4	2.400	0.01	4339	2.400	0.01	3254	2.400	0.01	2603	2.400	0.00	2169
9.6	9.601	0.01	1084	9.598	-0.02	813	9.601	0.01	650	9.592	-0.09	542
19.2	19.184	-0.08	542	19.195	-0.02	406	19.172	-0.15	325	19.219	0.10	270
57.6	57.551	-0.08	180	57.445	-0.27	135	57.339	-0.45	108	57.869	0.47	89
115.2	115.742	0.47	89	114.890	-0.27	67	115.741	0.47	53	115.739	0.47	44

波特率 (K)	SYNC = 0， BRG16 = 1， BRGH = 1 或 SYNC = 1， BRG16 = 1								
	Fosc = 13.889 MHz			Fosc = 6.250 MHz			Fosc = 4.167 MHz		
	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)	实际波特率 (K)	% 误差	SPBRG 值 (十进制)
0.3	0.300	0.00	11573	0.300	0.01	5207	0.300	-0.01	3472
1.2	1.200	-0.02	2893	1.200	0.01	1301	1.200	0.01	867
2.4	2.400	-0.02	1446	2.400	0.01	650	2.400	0.01	433
9.6	9.592	-0.08	361	9.586	-0.15	162	9.557	-0.44	108
19.2	19.184	-0.08	180	19.290	0.47	80	19.292	0.48	53
57.6	57.870	0.47	59	57.870	0.47	26	57.875	0.48	17
115.2	115.742	0.47	29	111.607	-3.12	13	115.750	0.48	8

20.1.3 自动波特率检测

增强型 USARTx 模块支持波特率自动检测和校准。此功能仅在异步模式下当 WUE 位清零时有效。

只要接收到起始位并且 ABDEN 位已置 1，就会开始自动波特率检测（图 20-1）。波特率计算采用自平均的方式。

在自动波特率检测（Auto-Baud Rate Detect，ABD）模式下， BRG 的时钟是反向的。不是由 BRG 为进入的 RXx 信号提供时钟信号，而是由 RXx 信号为 BRG 定时。在 ABD 模式下，内部波特率发生器被用作计数器来计算输入的串行字节流的位间隔时间。

一旦 ABDEN 位置 1，状态机就会将 BRG 清零并寻找起始位。为了正确计算比特率，自动波特率检测必须接收到一个值为 55h （ASCII 字符 U，也是 LIN 总线的同步字符）的字节。为了尽量减少输入信号不对称造成的影响，测量时段内要包含一个高位和一个低位时间。在起始位后，SPBRGx 使用预先选择的时钟源在 RXx 的第一个上升沿开始计数。在 RXx 引脚传输了 8 个位，或在检测到第 5 个上升沿后，会将相应 BRG 周期内的累加值保存在 SPBRGHx:SPBRGx 寄存器对中。当第 5 个时钟周期出现时（应与停止位对应），ABDEN 位会自动清零。

如果发生了 BRG 计满返回 （从 FFFFh 到 0000h 的溢出），会在 ABDOVF 状态位 （BAUDCONx<7>）有所反映。该位可在 BRG 溢出时由硬件置 1，也可以由用户通过软件置 1 或清零。在发生进位事件后，ABD 模式继续有效， ABDEN 位保持置 1 （图 20-2）。

在校准波特率周期时，BRG 寄存器时钟频率为预配置时钟频率的 1/8。请注意 BRG 时钟将由 BRG16 和 BRGH 位配置。不管 BRG16 如何设置，SPBRGx 和 SPBRGHx 都将被用作 16 位计数器。用户通过检查 SPBRGHx 寄存器的值是否为 00h，可以验证 8 位模式下是否发生了进位。表 20-4 所示为 BRG 计数器的时钟速率。

当产生 ABD 时序时， EUSARTx 状态机保持在空闲状态。一旦在 RXx 上检测到第 5 个上升沿，中断标志位 RCxIF 就会置 1。需要读取 RCREGx 中的值，来清除中断标志位 RCxIF。应丢弃 RCREGx 的值。

> **注 1：** 如果 WUE 位与 ABDEN 位同时置 1，自动波特率检测会在间隔字符*之后*的字节开始。
>
> **2：** 需要由用户来判断进入字符波特率是否处于所选 BRG 时钟源范围内。由于位错误率的原因，某些振荡频率和 EUSARTx 波特率的组合无法实现。使用自动波特率检测功能时，必须综合考虑系统总的时序和通信波特率。

表 20-4： BRG 计数器时钟速率

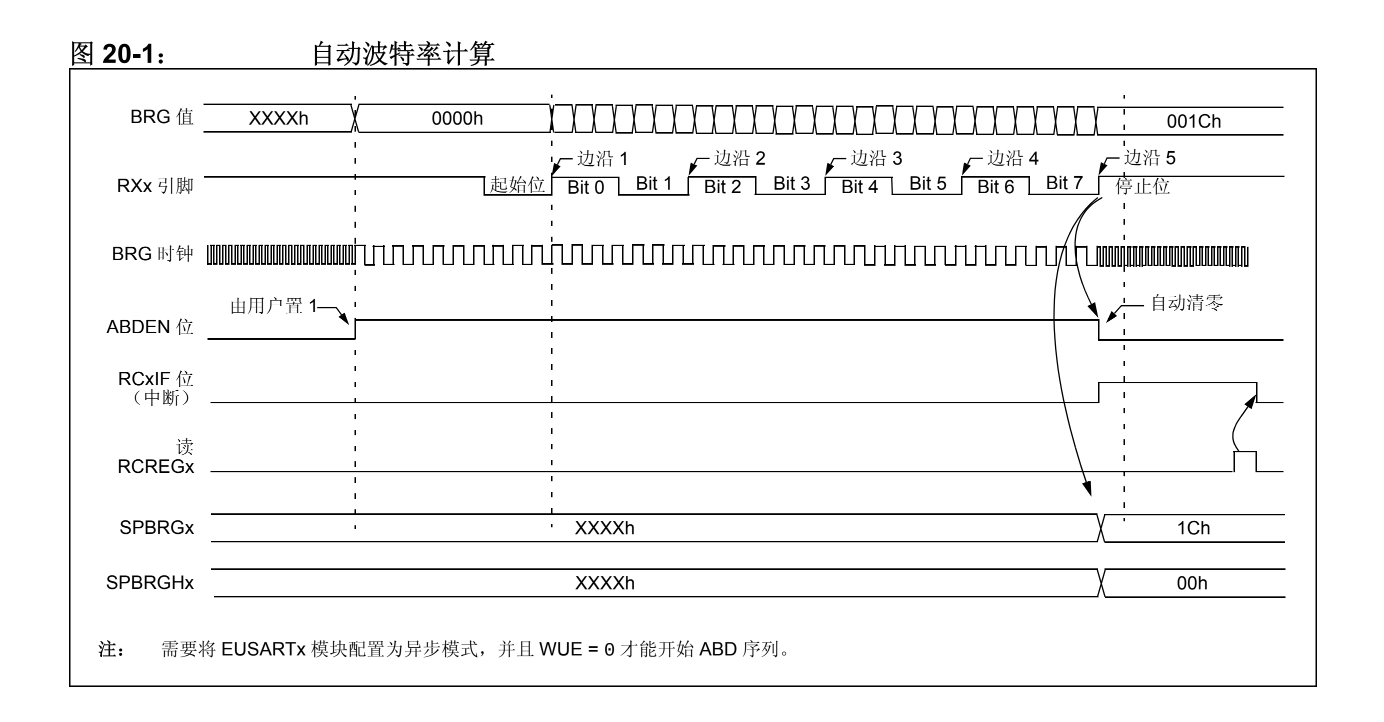
BRG16	BRGH	BRG 计数器时钟
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

注： 在产生 ABD 时序时，不管 BRG16 如何设置，SPBRGx 和 SPBRGHx 都被用作 16 位计数器。

20.1.3.1 ABD 和 EUSARTx 发送

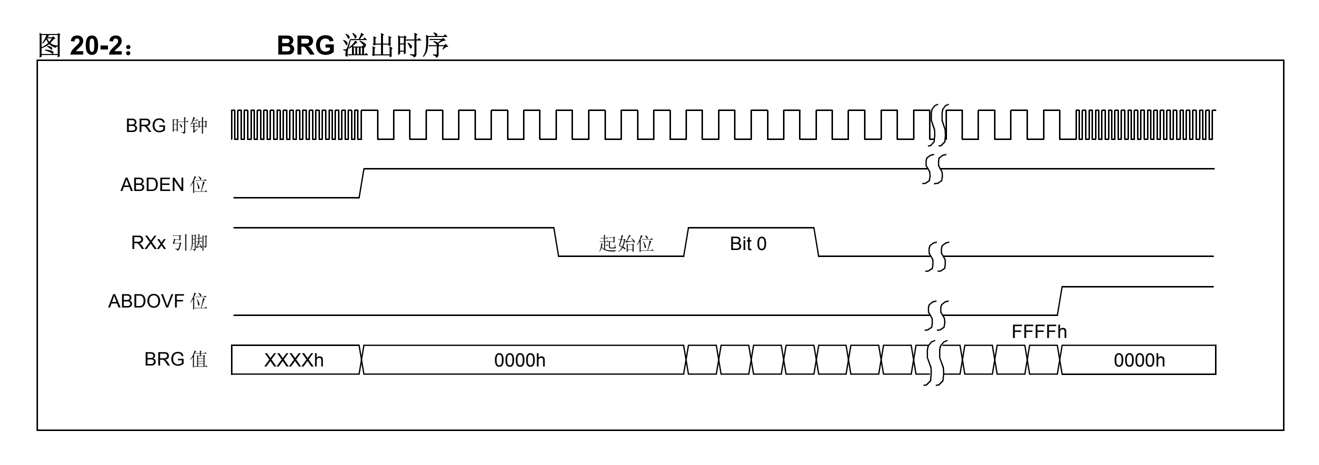
由于 ABD 采样期间 BRG 时钟是反向的，因此在 ABD 期间不能使用 EUSARTx 发送器。这意味着只要 ABDEN 位置 1，就不能写入 TXREGx。用户还应确保在发送期间 ABDEN 不能为置 1 状态，否则可能会导致无法预料的 EUSARTx 操作。

图 20-1： 自动波特率计算



注： 需要将 EUSARTx 模块配置为异步模式，并且 WUE = 0 才能开始 ABD 序列。

图 20-2： BRG 溢出时序



 超前信息

20.2 EUSARTx 异步模式

通过将 SYNC 位 （TXSTAx<4>）清零可选择异步工作模式。在此模式下， EUSARTx 使用标准的不归零（Non-Return-to-Zero，NRZ）格式（1 个起始位、8 个或 9 个数据位和 1 个停止位）。最常用的格式是含有 8 个数据位。片上专用8位/16位波特率发生器可借助于振荡器产生标准波特率频率。

EUSARTx 首先发送和接收 LSb。 EUSARTx 模块的发送器和接收器在功能上是独立的，但采用相同的数据格式和波特率。波特率发生器可以根据BRGH位和BRG16位（TXSTAx<2> 和 BAUDCONx<3>）的设置值产生两种不同的波特率时钟，频率分别为移位速率的 16 倍或 64 倍。硬件不支持奇偶校验，但可以用软件实现，校验值保存在第 9 个数据位中。

TXCKP（BAUDCONx<4>）和 RXDTP（BAUDCONx<5>）位可使得 TXx 和 RXx 信号反相 （极性相反）。在 TTL 和 RS-232 电平之间缓冲信号的器件也能反转信号。将 TXCKP 和 RXDTP 位置 1 允许使用提供缓冲的电路，但不会反转信号。

当工作在异步模式下时， EUSARTx 模块包括以下重要组成部分：

- 波特率发生器
- 采样电路
- 异步发送器
- 异步接收器
- 同步间隔字符自动唤醒
- 12 位间隔字符发送
- 自动波特率检测

20.2.1 EUSARTx 异步发送器

图 20-3 给出了 EUSARTx 发送器框图。发送器的核心是发送（串行）移位寄存器（Transmit Shift Register，TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREGx 中获取数据。TXREGx 寄存器中的数据由软件写入。在前一次装入数据的停止位发送前，不会向TSR寄存器装入新数据。一旦停止位发送完毕，TXREGx 寄存器中的新数据 （如果有）就会被装入 TSR。

一旦 TXREGx 寄存器向 TSR 寄存器传输了数据 （在 1 个 T_{CY} 内发生）， TXREGx 寄存器就为空，同时 TXxIF 标志位置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许或禁止该中断。不管 TXxIE 的状态如何，只要中断发生， TXxIF 就会置 1 并且不能用软件清零。 TXxIF 不会在TXREGx装入新数据时立即被清零，而是在装入指令后的第二个指令周期被清零。因此在TXREGx装入新数据后立即查询 TXxIF，会得到无效结果。

TXxIF 指示的是 TXREGx 寄存器的状态，而另一个位 TRMT （TXSTAx<1>）则指示 TSR 寄存器的状态。TRMT是只读位，它在TSR寄存器为空时被置1。TRMT 位与任何中断均无关联，因此要确定TSR寄存器是否为空，用户只能对此位进行查询。

注	1：	TSR 寄存器并未映射到数据存储器中，因此用户无法直接访问它。
	2：	当使能位TXEN置1时，标志位TXxIF置1。

设置异步发送操作的步骤如下：

1. 对SPBRGHx:SPBRGx寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1，使能异步串口。
3. 如果要使 TXx 引脚的信号反相，将 TXCKP 位置 1。
4. 如果需要中断，将允许位 TXxIE 置 1。
5. 如果需要 9 位发送，将发送位 TX9 置 1。可以作为地址 / 数据位使用。
6. 通过将 TXEN 位置 1 使能发送，此操作同时也会将 TXxIF 位置 1。
7. 如果选择了 9 位发送，将第 9 位装入 TX9D 位。
8. 将数据装入 TXREGx 寄存器 （开始发送）。
9. 如果使用中断，应确保 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）已置 1。

图 20-3：　　EUSARTx 发送框图

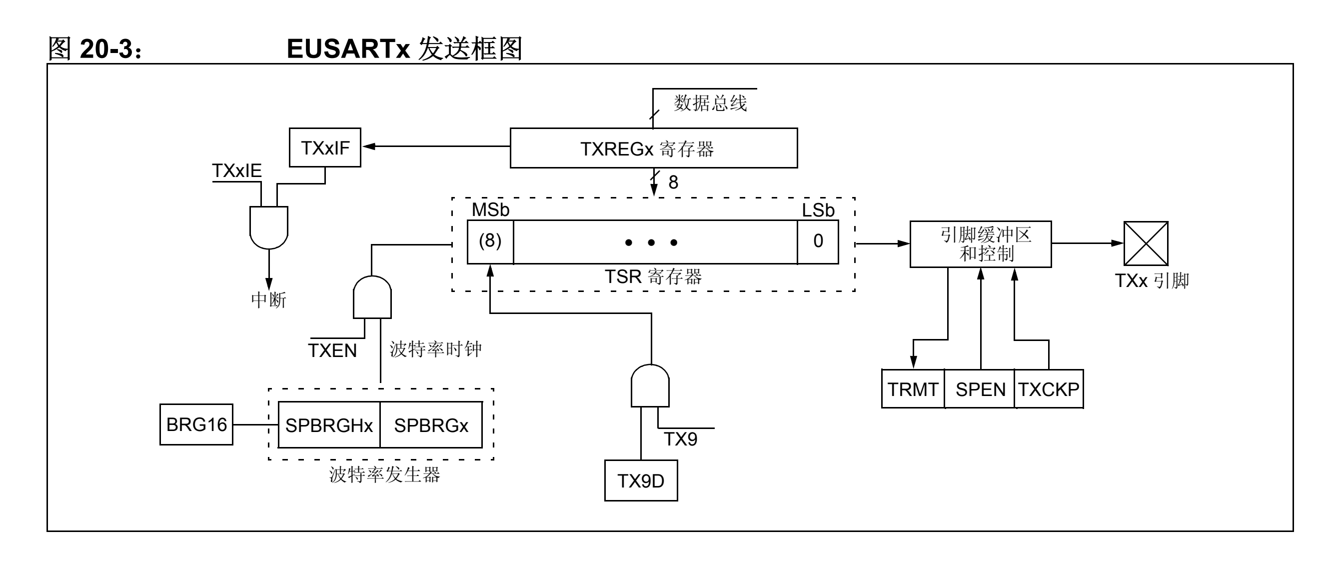


图 20-4：　　异步发送， TXCKP = 0 （TXx 未反相）

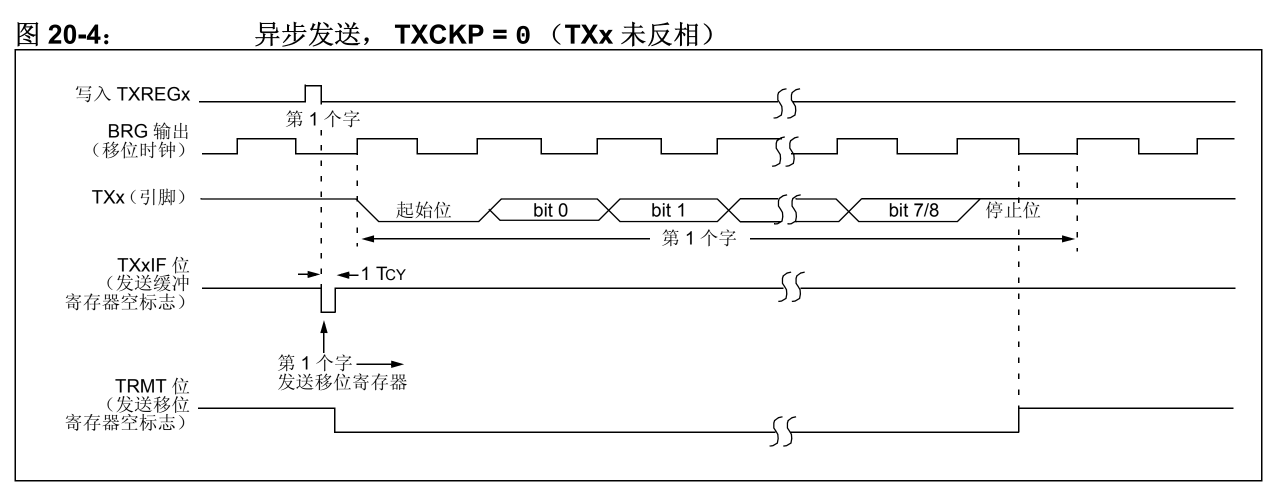


图 20-5：　　异步发送 （背对背）， TXCKP = 0 （TXx 未反相）

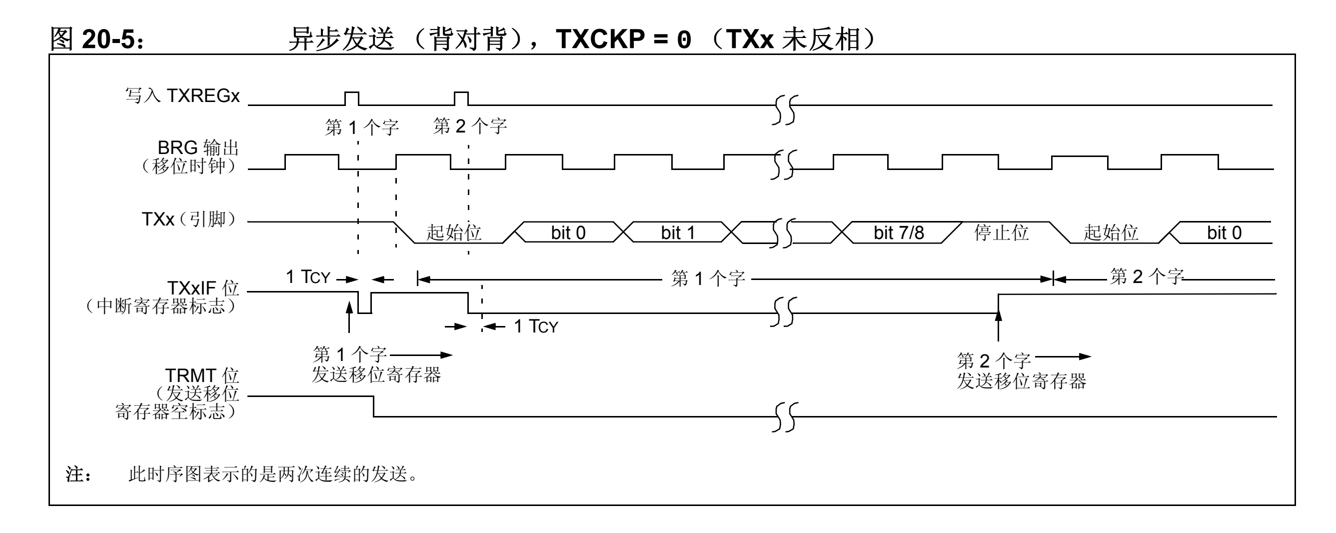


表 **20-5**： 与异步发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF(1)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE(1)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP(1)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREGx	EUSARTx 发送寄存器								61
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用单元，读为 0。异步发送不使用阴影单元。

注 1： 这些位仅在 80 引脚和 100 引脚器件上实现；在其他器件上则未实现且读为 0。

20.2.2 EUSARTx 异步接收器

图 20-6 给出了接收器框图。在 RXx 引脚上接收数据，并驱动数据恢复电路。数据恢复电路实际上是一个工作频率为 16 倍波特率的高速移位器，而主接收串行移位器的工作频率等于比特率或 FOSC。此模式通常用于 RS-232 系统。

RXDTP 位（BAUDCON<5>）可使得 RXx 信号反相（极性相反）。从 RS-232 到 TTL 电平之间缓冲信号的器件也可执行信号的反转（当 RS-232 = 正， TTL = 0 时）。通过将 RXDTP 置 1 反转 RXx 引脚数据的极性可允许用户使用只提供缓冲并不反转信号的电路。

设置异步接收操作的步骤如下：

1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1，使能异步串口。
3. 如果要使 RXx 引脚的信号反相，将 RXDTP 位置 1。
4. 如果需要中断，将允许位 RCxIE 置 1。
5. 如果需要接收 9 位数据，将 RX9 位置 1。
6. 通过将 CREN 位置 1 使能接收。
7. 当接收完成时标志位 RCxIF 将被置 1，此时如果中断允许位RCxIE已置1，还将产生一个中断。
8. 读 RCSTAx 寄存器获取第 9 位数据 （如果已使能），并判断在接收过程中是否发生了错误。
9. 读 RCREGx 寄存器来读取接收到的 8 位数据。
10. 如果发生错误，通过将使能位 CREN 清零来清除错误。
11. 如果使用中断，应确保 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）已置 1。

20.2.3 设置带有地址检测功能的 9 位模式

此模式通常用于 RS-485 系统。按如下步骤设置带有地址检测功能的异步接收操作：

1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1，使能异步串口。
3. 如果要使 RXx 引脚的信号反相，将 RXDTP 位置 1。如果要使 TXx 引脚的信号反相，将 TXCKP 位置 1。
4. 如果需要中断，将 RCEN 位置 1 并用 RCxIP 位选择所需的优先级别。
5. 将 RX9 位置 1，使能 9 位接收。
6. 将 ADDEN 位置 1，使能地址检测。
7. 将 CREN 位置 1，使能接收。
8. 当接收完成时 RCxIF 位将被置 1。此时如果 RCxIE 和 GIE 位已置 1，还将响应中断。
9. 读 RCSTAx 寄存器判断在接收时是否发生了错误，同时读取第 9 位数据 （如果适用）。
10. 读 RCREGx 判断是否正在对器件进行寻址。
11. 如果发生错误，将 CREN 位清零。
12. 如果已找到器件，将 ADDEN 位清零，允许所有的接收数据进入接收缓冲区并中断 CPU。

 超前信息

图 20-6： EUSARTx 接收框图

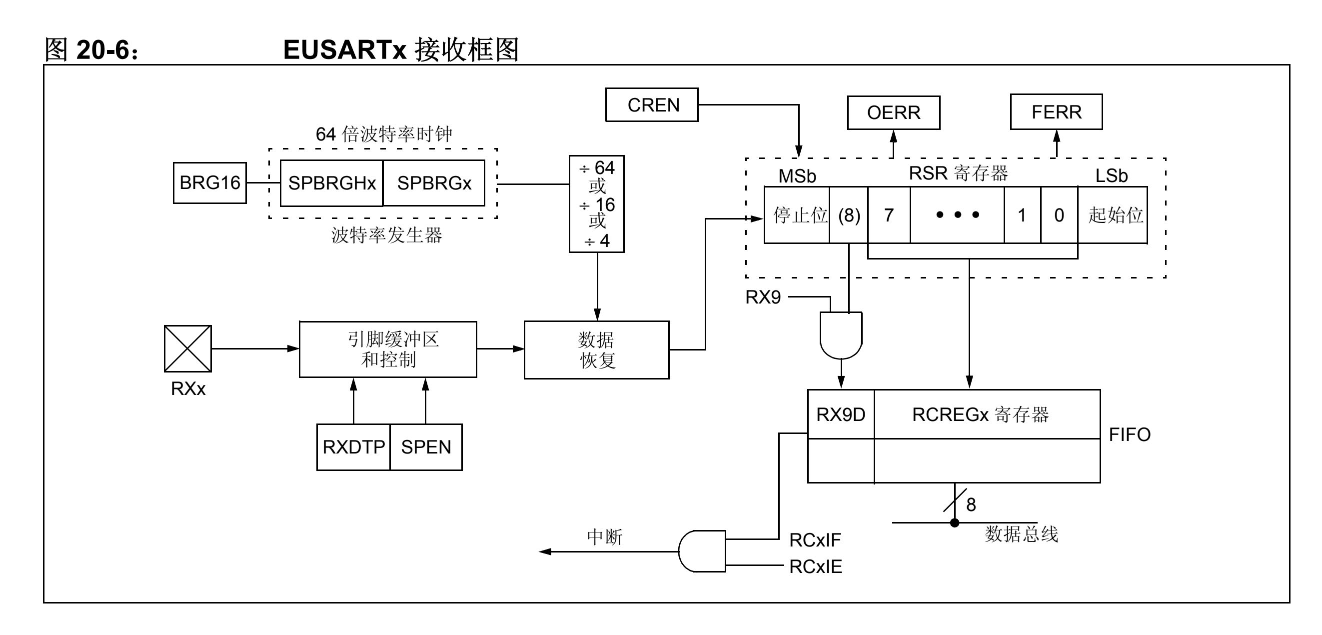
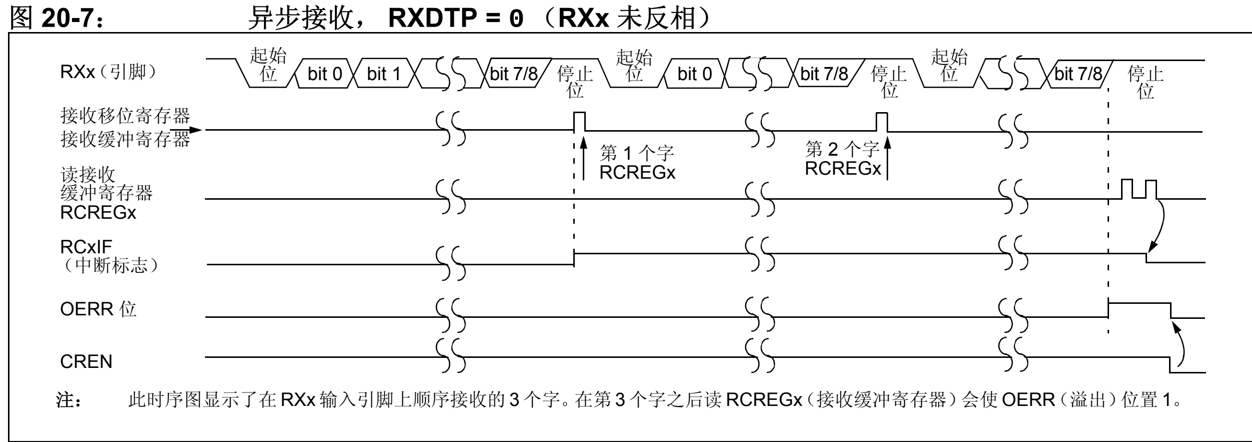


图 20-7： 异步接收， RXDTP = 0 （RXx 未反相）



注： 此时序图显示了在 RXx 输入引脚上顺序接收的 3 个字。在第 3 个字之后读 RCREGx（接收缓冲寄存器）会使 OERR（溢出）位置 1。

表 20-6： 与异步接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF(1)	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE(1)	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP(1)	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREGx	EUSARTx 接收寄存器								61
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用单元，读为 0。异步接收不使用阴影单元。

注 1： 这些位仅在 80 引脚和 100 引脚器件上实现；在其他器件上则未实现且读为 0。

20.2.4 同步间隔字符自动唤醒

在休眠模式下，EUSARTx 的所有时钟都会暂停。因此，波特率发生器处于非激活状态，且无法进行正确的数据接收。自动唤醒功能允许当RXx/DTx线上有事件发生时唤醒控制器，该功能需要 EUSARTx 工作在异步模式下。

通过将 WUE 位（BAUDCONx<1>）置 1，使能自动唤醒功能。该功能启用后，将禁止 RXx/DTx 上的典型接收操作，且EUSARTx保持在空闲状态并监视唤醒事件（不管 CPU 运行模式如何）。唤醒事件是指 RXx/DTx 线上发生高电平到低电平的转换。（这刚好与 “同步间隔” 字符或 LIN 协议唤醒信号字符的启动条件一致。）

唤醒事件后，模块产生一个 RCxIF 中断。在正常工作模式下，中断会与 Q 时钟同步产生（图 20-8）；如果器件处于休眠模式，则两者是不同步的（图 20-9）。通过读 RCREGx 寄存器可清除中断条件。

唤醒事件后，当 RXx 线上出现由低向高的电平转换时，WUE 位自动清零。此时，EUSARTx 模块将从空闲状态返回正常工作模式，由此用户可知 “同步间隔” 事件已经结束。

20.2.4.1 使用自动唤醒功能的特别注意事项

因为自动唤醒功能是通过检测RXx/DTx上的上升沿跳变实现的，所以在停止位前该引脚上任何的状态改变都可能会产生错误的结束信号并导致数据或帧错误。因此，为了确保正常的传输，必须首先发送全 0 字符。对于标准的RS-232 器件，该字符是 00h （8 位），而对于 LIN 总线器件则是 000h （12 位）。

另外还必须考虑振荡器起振时间，尤其在采用起振时间较长的振荡器 （即， HS 或 HSPLL 模式）应用中更要注意这一点。“同步间隔”（或唤醒信号）字符必须足够长，并且跟有足够长的时间间隔，以便使选定振荡器有充足的时间起振并保证 EUSARTx 正确初始化。

20.2.4.2 使用 WUE 位的特别注意事项

使用 WUE 和 RCxIF 事件的时序来判断接收数据的有效性时，有可能会引起一些混淆。如前所述，将 WUE 位置 1 会使 EUSARTx 进入空闲状态。唤醒事件会通过将 RCxIF 位置 1 产生一个接收中断。此后当 RXx/DTx 出现上升沿时，WUE 位被清零。然后通过读 RCREGx 寄存器清除中断条件。一般情况下，RCREGx 中的数据是无效数据，应该丢弃。

WUE 位清零 （或仍然置 1）同时 RCxIF 标志位置 1 并不能表明 RCREGx 中数据接收是完整的。用户应该考虑在固件中同时验证是否完整地接收了数据。

要确保没有丢失有效数据，应检查RCIDL位来验证是否还在接收数据。如果不在接收数据，则可将 WUE 位置 1，使器件立即进入休眠模式。

 超前信息

图 20-8：　　正常工作模式下的自动唤醒位 （WUE）时序

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4

OSC1

WUE 位 (1)

由用户置 1

自动清零

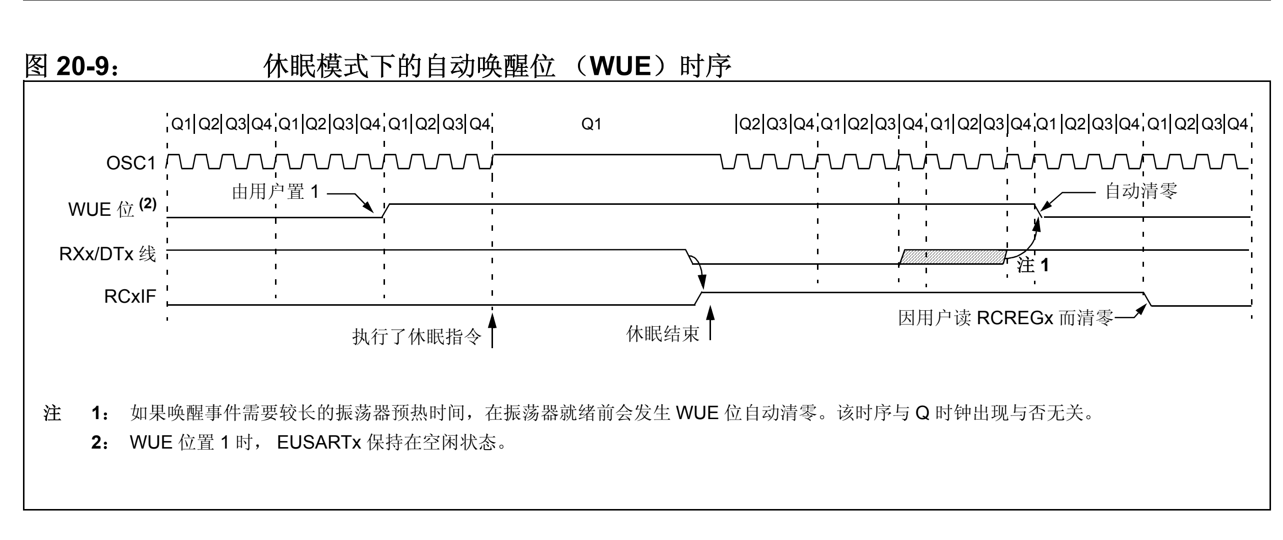
RXx/DTx 线

RCxIF

因用户读 RCREGx 而清零

注　1：　WUE 位置 1 时， EUSARTx 保持在空闲状态。

图 20-9：　　休眠模式下的自动唤醒位 （WUE）时序



注　1：　如果唤醒事件需要较长的振荡器预热时间，在振荡器就绪前会发生 WUE 位自动清零。该时序与 Q 时钟出现与否无关。
　　2：　WUE 位置 1 时， EUSARTx 保持在空闲状态。

20.2.5 间隔字符序列

EUSARTx 模块能够发送符合 LIN 总线标准的特殊间隔字符序列。发送的间隔字符包括 1 个起始位，后面跟有 12 个 0 位和一个停止位。当发送移位寄存器装有数据时，只要 SENDB 和 TXEN 位（TXSTAx<3> 和 TXSTAx<5>）置 1，就会发送帧间隔字符。请注意写入 TXREGx 的数据值会被忽略，并会发送全 0。

在发送了相应的停止位后，硬件会自动将 SENDB 位复位。这样用户可以在间隔字符（在 LIN 规范中通常是同步字符）后预先将下一个要发送字节装入发送 FIFO 队列。

请注意在发送间隔字符时写入TXREGx的数据值会被忽略。写入仅仅是为了启动正确的序列。

正如其在正常发送操作中一样，TRMT 位表明发送正在进行还是处于空闲状态。关于发送间隔字符的时序，请参见图 20-10。

20.2.5.1 间隔和同步发送序列

下述序列会发送一个报文帧头，包括一个间隔字符和其后的自动波特率同步字节。此序列适用于典型的 LIN 总线主器件。

1. 将 EUSARTx 配置为所需的模式。
2. 将 TXEN 和 SENDB 位置 1，以设置间隔字符。
3. 将无效字符装入 TXREGx，启动发送（该值会被忽略）。
4. 将55h写入TXREGx，以便把同步字符装入FIFO 缓冲区。
5. 间隔字符发送后，硬件会将 SENDB 位复位。此时，同步字符会以预先配置的模式发送。

当 TXREGx 为空时 （由 TXxIF 指出），下一个数据字节会写入 TXREGx。

20.2.6 接收间隔字符

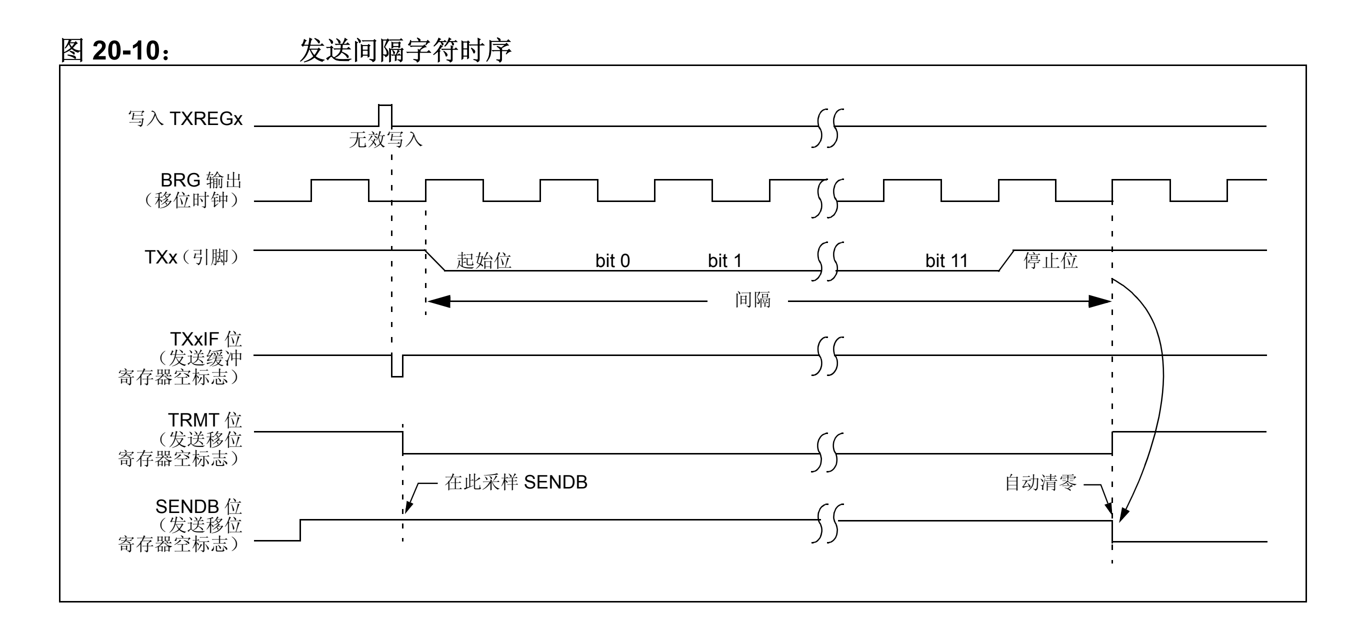
增强型 USARTx 模块接收间隔字符有两种方法。

第一种方法是强制将波特率配置为典型速率的 9/13。这可以使停止位在正确的采样点（对于间隔字符为起始位之后的 13 位，对于典型数据则是 8 个数据位）产生。

第二种方法是使用**第 20.2.4 节 "同步间隔字符自动唤醒"**中描述的自动唤醒功能。通过使能此功能，EUSARTx 将采样 RXx/DTx 线的下两次跳变，产生一个 RCxIF 中断，接收下一个数据字节，并在随后产生另一个中断。

请注意在间隔字符后，用户通常希望使能自动波特率检测功能。无论使用哪种方法，用户都可以在检测到 TXxIF 中断时马上将 ABDEN 位置 1。

图 20-10： 发送间隔字符时序



 超前信息

20.3 EUSARTx 同步主控模式

将 CSRC 位 （TXSTAx<7>）置 1 可以进入同步主控模式。在此模式中，数据以半双工方式发送（即发送和接收不能同时进行）。发送数据时禁止接收，反之亦然。将 SYNC 位 （TXSTAx<4>）置 1 可以进入同步模式。此外，应将使能位 SPEN （RCSTAx<7>）置 1，分别把 TXx 和 RXx 引脚配置为 CKx（时钟）和 DTx（数据）线。

时钟极性（CKx）是通过 TXCKP 位（BAUDCON<4>）选择的。将 TXCKP 置 1 是将空闲状态的 CKx 设为高电平，将该位清零则将空闲状态的 CKx 设为低电平。数据极性（DTx）是通过 RXDTP 位（BAUDCONx<5>）选择的。将 RXDTP 置 1 是将空闲状态的 DTx 设为高电平，将该位清零则将空闲状态的 DTx 设为低电平。当 CKx 返回到其空闲状态时对 DTx 进行采样。此选项支持将本模块与 Microwire 器件配合使用。

20.3.1 EUSARTx 同步主控发送

图 20-3 给出了 EUSARTx 发送器框图。发送器的核心是发送（串行）移位寄存器（TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREGx 中获取数据。 TXREGx 寄存器中的数据由软件写入。在前一次装入数据的最后一位发送完成后，才向 TSR 寄存器装入新数据。一旦最后一位发送完成，就会将 TXREGx 寄存器的新数据（如果有）装入 TSR。

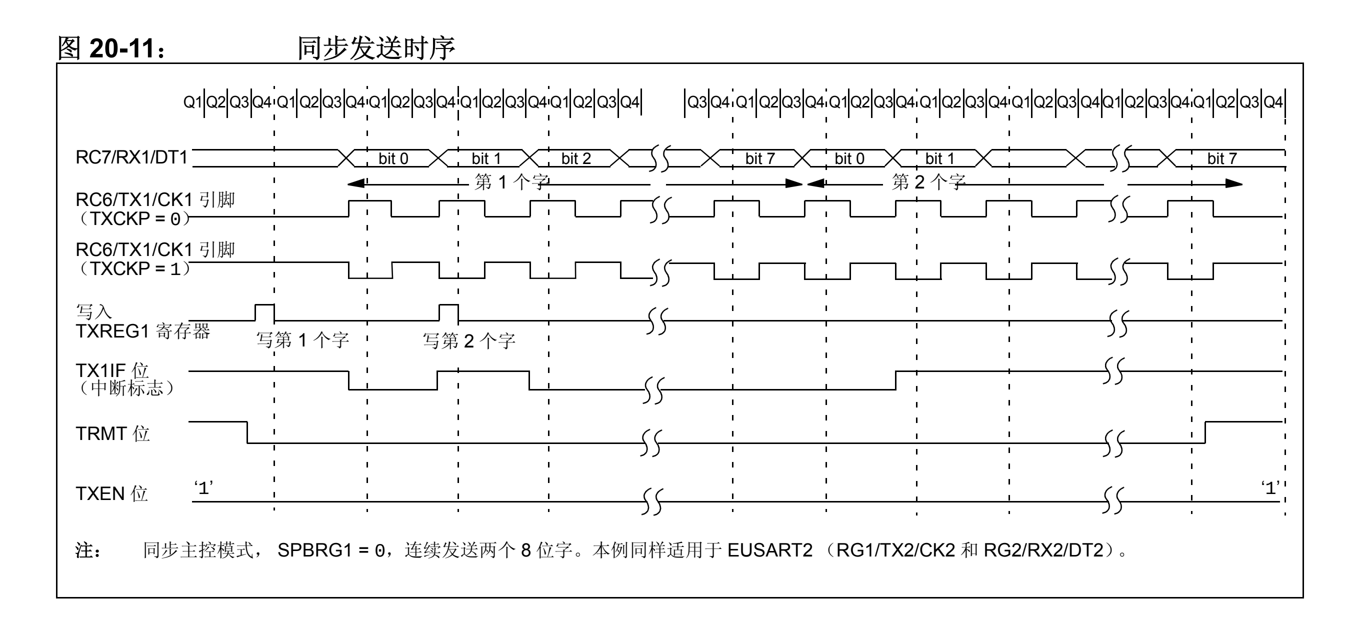
一旦 TXREGx 寄存器向 TSR 寄存器传输了数据（在 1 个 TCY 内发生）， TXREGx 寄存器就为空，同时 TXxIF 标志位被置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许或禁止该中断。TXxIF 的设置不受 TXxIE 状态的影响，且不能用软件清零。只有在新数据写入 TXREGx 寄存器时， TXxIF 才会复位。

标志位 TXxIF 表示的是 TXREGx 寄存器的状态，而另一个标志位 TRMT （TXSTAx<1>）则表示TSR 寄存器的状态。TRMT 位是一个只读位，当 TSR 为空时，TRMT 被置1。TRMT 位与任何中断均无关联，因此要确定 TSR 寄存器是否为空，用户只能对此位进行查询。TSR 并未映射到数据存储器中，所以用户不能直接访问它。

设置同步主控发送操作的步骤如下：

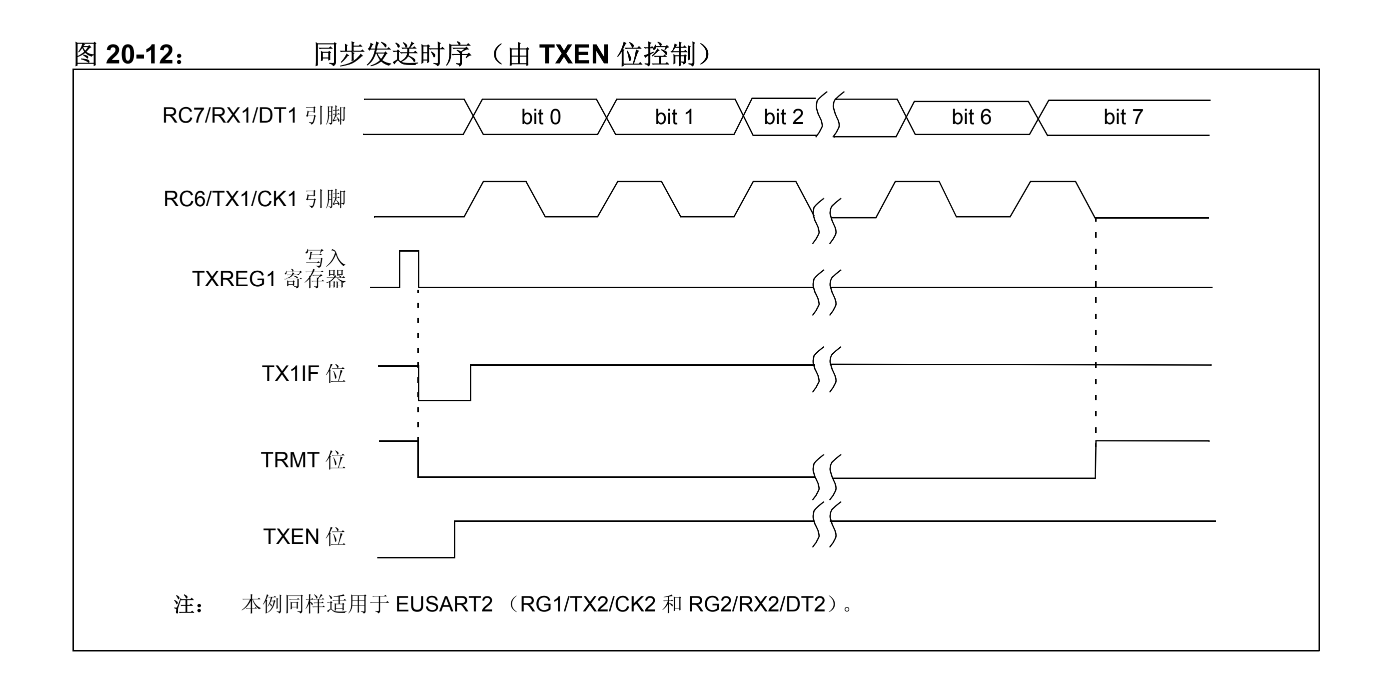
1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 如果要使 CKx 引脚的信号反相，将 TXCKP 位置 1。如果要使 DTx 引脚的信号反相，将 RXDTP 位置 1。
4. 如果需要中断，将允许位 TXxIE 置 1。
5. 如果需要 9 位发送，将 TX9 位置 1。
6. 将 TXEN 位置 1，使能发送。
7. 如果选择了 9 位发送，将第 9 位装入 TX9D 位。
8. 将数据装入 TXREGx 寄存器，启动发送。
9. 如果使用中断，应确保 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）已置 1。

图 20-11： 同步发送时序



注： 同步主控模式， SPBRG1 = 0，连续发送两个 8 位字。本例同样适用于 EUSART2 （RG1/TX2/CK2 和 RG2/RX2/DT2）。

图 20-12： 同步发送时序 （由 **TXEN** 位控制）



注： 本例同样适用于 EUSART2 （RG1/TX2/CK2 和 RG2/RX2/DT2）。

表 20-7： 与同步主控发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF(1)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE(1)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP(1)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREGx	EUSARTx 发送寄存器								61
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用，读为 0。同步主控发送不使用阴影单元。

注 1：这些位仅在 80 引脚和 100 引脚器件上实现；在其他器件上则未实现且读为 0。

 超前信息

20.3.2 EUSARTx 同步主控接收

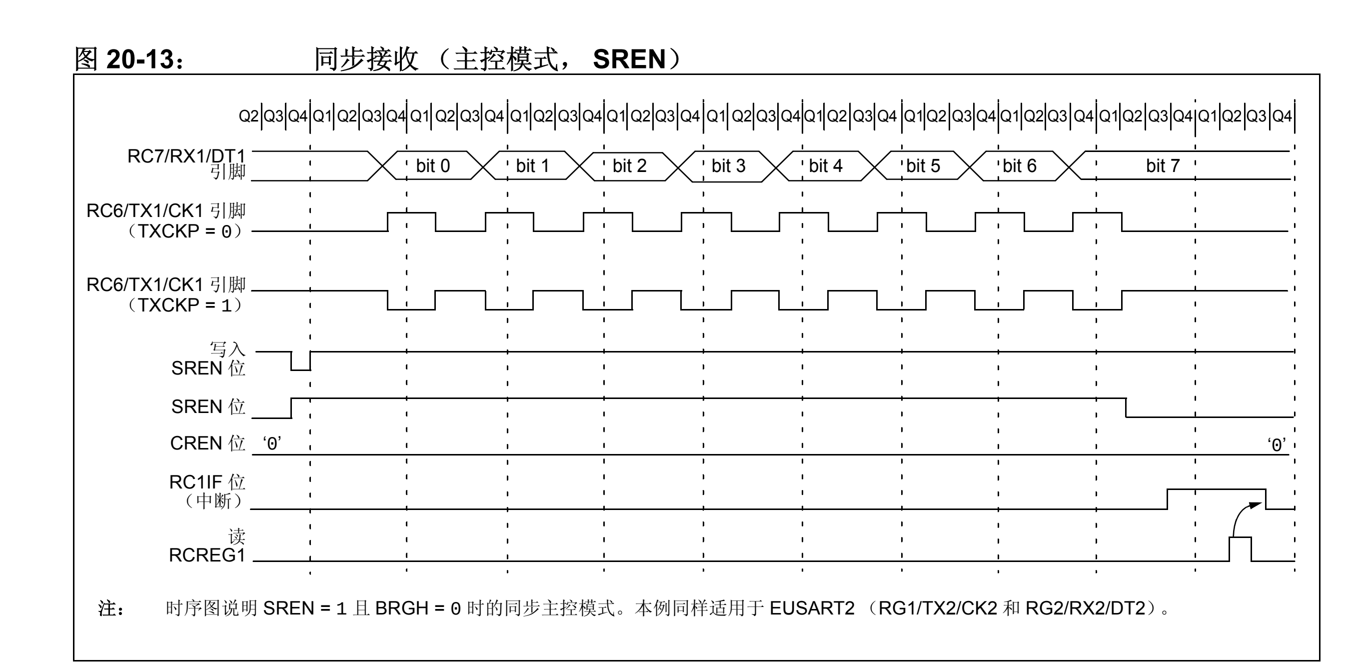
一旦选择了同步模式，只要将单字节接收使能位 SREN（RCSTAx<5>）或连续接收使能位 CREN（RCSTAx<4>）置 1，即可使能接收。在时钟的下降沿采样 RXx 引脚上的数据。

如果使能位 SREN 置 1，则只接收单个字。如果将使能位 CREN 置 1，则会连续接收数据，直到将 CREN 位清零。如果两个位均被置 1，则 CREN 优先。

设置同步主控接收操作的步骤如下：

1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 确保将 CREN 和 SREN 位清零。
4. 如果要使 CKx 引脚的信号反相，将 TXCKP 位置 1。如果要使 DTx 引脚的信号反相，将 RXDTP 位置 1。
5. 如果需要中断，将允许位 RCxIE 置 1。
6. 如果需要接收 9 位数据，将 RX9 位置 1。
7. 如果需要单字节接收，将 SREN 位置 1。如果需要连续接收，将 CREN 位置 1。
8. 当接收完成时中断标志位 RCxIF 将置 1，此时如果中断允许位 RCxIE 已置 1，则还将产生一个中断。
9. 读 RCSTAx 寄存器获取第 9 位数据（如果已使能），并判断在接收过程中是否发生了错误。
10. 读 RCREGx 寄存器来读取接收到的 8 位数据。
11. 如果发生错误，将 CREN 位清零以清除错误。
12. 如果使用中断，应确保 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）已置 1。

图 20-13： 同步接收（主控模式， **SREN**）



注： 时序图说明 SREN = 1 且 BRGH = 0 时的同步主控模式。本例同样适用于 EUSART2（RG1/TX2/CK2 和 RG2/RX2/DT2）。

表 20-8： 与同步主控接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF(1)	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE(1)	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP(1)	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREGx	EUSARTx 接收寄存器								61
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用，读为 0。同步主控接收不使用阴影单元。
注 1：这些位仅在 80 引脚和 100 引脚器件上实现；在其他器件上则未实现且读为 0。

20.4 EUSARTx 同步从动模式

将 CSRC（TXSTAx<7>）清零可进入同步从动模式。此模式与同步主控模式的区别在于移位时钟由 CKx 引脚上的外部时钟提供 （主控模式中由内部时钟提供）。这使得器件能在任何低功耗模式下发送或接收数据。

20.4.1 EUSARTx 同步从动发送

除了休眠模式以外，同步主控模式和从动模式的工作方式是相同的。

如果向 TXREGx 写 2 个字，然后执行 SLEEP 指令，则会发生以下事件：

a） 第一个字立即传送到 TSR 寄存器进行发送。
b） 第二个字仍保留在 TXREGx 寄存器中。
c） 不会将标志位 TXxIF 置 1。
d） 当第一个字移出 TSR 后，TXREGx 寄存器将把第二个字传送给 TSR，同时将标志位 TXxIF 置 1。
e） 如果中断允许位 TXxIE 置 1，中断将把器件从休眠状态唤醒。如果允许了全局中断，程序则会跳转到中断向量处执行。

设置同步从动发送的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1、CSRC 位清零，使能同步从动串口。
2. 将 CREN 和 SREN 位清零。
3. 如果要使 CKx 引脚的信号反相，将 TXCKP 位置 1。如果要使 DTx 引脚的信号反相，将 RXDTP 位置 1。
4. 如果需要中断，将允许位 TXxIE 置 1。
5. 如果需要 9 位发送，将 TX9 位置 1。
6. 将使能位 TXEN 置 1 以使能发送。
7. 如果选择了 9 位发送，将第 9 位装入 TX9D 位。
8. 将数据装入 TXREGx 寄存器，启动发送。
9. 如果使用中断，应确保 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）已置 1。

表 20-9： 与同步从动发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF(1)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE(1)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP(1)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREGx	EUSARTx 发送寄存器								61
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用，读为 0。同步从动发送不使用阴影单元。
注 1：这些位仅在 80 引脚和 100 引脚器件上实现；在其他器件上则未实现且读为 0。

20.4.2 EUSARTx 同步从动接收

除了休眠模式、空闲模式以及在从动模式下忽略 SREN 位以外，同步主控和从动模式的工作方式完全相同。

如果在进入休眠或空闲模式前将 CREN 位置 1 使能接收，那么在低功耗模式下可以接收到一个数据字。接收到该字后，RSR 寄存器将把数据发送到 RCREGx 寄存器。如果中断允许位 RCxIE 已置 1，产生的中断将把芯片从低功耗模式唤醒。如果允许了全局中断，程序则会跳转到中断向量处执行。

设置同步从动接收操作的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零，使能同步从动串口。
2. 如果需要中断，将允许位 RCxIE 置 1。
3. 如果要使 CKx 引脚的信号反相，将 TXCKP 位置 1。如果要使 DTx 引脚的信号反相，将 RXDTP 位置 1。
4. 如果需要接收 9 位数据，将 RX9 位置 1。
5. 将使能位 CREN 置 1 以使能接收。
6. 当接收完成时 RCxIF 位将被置 1。如果中断允许位 RCxIE 置 1，还将产生一个中断。
7. 读 RCSTAx 寄存器获取第 9 位数据（如果已使能），并判断在接收过程中是否发生了错误。
8. 读 RCREGx 寄存器来读取接收到的 8 位数据。
9. 如果发生错误，将 CREN 位清零以清除错误。
10. 如果使用中断，应确保 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）已置 1。

表 20-10： 与同步从动接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR3	SSP2IF	BCL2IF	RC2IF(1)	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	61
PIE3	SSP2IE	BCL2IE	RC2IE(1)	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	61
IPR3	SSP2IP	BCL2IP	RC2IP(1)	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	61
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREGx	EUSARTx 接收寄存器								61
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	62
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								62
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								62

图注： — = 未用，读为 0。同步从动接收不使用阴影单元。

注 1：这些位仅在 80 引脚和 100 引脚器件上实现；在其他器件上则未实现且读为 0。

 超前信息

21.0 10 位模数转换器 （A/D）模块

64 引脚器件的模数 （Analog-to-Digital， A/D）转换器模块有 11 路输入，80 引脚器件有 15 路输入，100 引脚器件有 16 路输入。此模块能将一个模拟输入信号转换成相应的 10 位数字信号。

此模块有五个寄存器：

- A/D 转换结果高位寄存器 （ADRESH）
- A/D 转换结果低位寄存器 （ADRESL）
- A/D 转换控制寄存器 0 （ADCON0）
- A/D 转换控制寄存器 1 （ADCON1）
- A/D 转换控制寄存器 2 （ADCON2）

ADCON0 寄存器（如寄存器 21-1 所示）控制 A/D 模块的工作。ADCON1 寄存器（如寄存器 21-2 所示）配置端口引脚功能。 ADCON2 寄存器 （如寄存器 21-3 所示）配置 A/D 时钟源、可编程采样时间和输出结果的对齐方式。

寄存器 21-1： ADCON0：A/D 控制寄存器 0

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **ADCAL**：A/D 校准位
1 = 在下个 A/D 转换时执行校准
0 = 正常 A/D 转换器操作 （不执行转换）

bit 6 未用：读为 0

bit 5-2 **CHS3:CHS0**：模拟通道选择位
0000 = 通道 0 （AN0）
0001 = 通道 1 （AN1）
0010 = 通道 2 （AN2）
0011 = 通道 3 （AN3）
0100 = 通道 4 （AN4）
0101 = 通道 5 （AN5）[1,3]
0110 = 通道 6 （AN6）
0111 = 通道 7 （AN7）
1000 = 通道 8 （AN8）
1001 = 通道 9 （AN9）
1010 = 通道 10 （AN10）
1011 = 通道 11 （AN11）
1100 = 通道 12 （AN12）[2,3]
1101 = 通道 13 （AN13）[2,3]
1110 = 通道 14 （AN14）[2,3]
1111 = 通道 15 （AN15）[2,3]

bit 1 **GO/$\overline{\text{DONE}}$**：A/D 转换状态位
当 ADON = 1 时：
1 = A/D 转换正在进行
0 = A/D 空闲

bit 0 **ADON**：A/D 模块使能位
1 = 使能 A/D 转换器模块
0 = 禁止 A/D 转换器模块

注 1： 该通道仅在 100 引脚器件上实现。
2： 这些通道仅在 80 引脚和 100 引脚器件上实现。
3： 在未实现通道上执行转换会返回随机值。

寄存器 21-2： ADCON1：A/D 控制寄存器 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	VCFG0	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-6 未用：读为 0

bit 5 **VCFG0**：参考电压配置位 （V_{REF-} 电压源）
1 = V_{REF-} (AN2)
0 = AV_{SS}

bit 4 **VCFG0**：参考电压配置位 （V_{REF+} 电压源）
1 = V_{REF+} (AN3)
0 = AV_{DD}

bit 3-0 **PCFG3:PCFG0**：A/D 端口配置控制位：

PCFG3: PCFG0	AN15[1]	AN14[1]	AN13[1]	AN12[1]	AN11	AN10	AN9	AN8	AN7	AN6	AN5[2]	AN4	AN3	AN2	AN1	AN0
0000	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A
0111	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

A = 模拟输入 D = 数字 I/O

注 **1：** AN12 到 AN15 仅在 80 引脚和 100 引脚器件上可用。
2： AN5 仅在 100 引脚器件上可用。

寄存器 21-3： ADCON2：A/D 控制寄存器 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **ADFM**：A/D 结果格式选择位
1 = 右对齐
0 = 左对齐

bit 6 未用：读为 0

bit 5-3 **ACQT2:ACQT0**：A/D 采样时间选择位
111 = 20 个 T_{AD}
110 = 16 个 T_{AD}
101 = 12 个 T_{AD}
100 = 8 个 T_{AD}
011 = 6 个 T_{AD}
010 = 4 个 T_{AD}
001 = 2 个 T_{AD}
000 = 0 个 T_{AD}[1]

bit 2-0 **ADCS2:ADCS0**：A/D 转换时钟选择位
111 = F_{RC}（时钟来自 A/D 模块 RC 振荡器）[1]
110 = $F_{OSC}/64$
101 = $F_{OSC}/16$
100 = $F_{OSC}/4$
011 = F_{RC}（时钟来自 A/D 模块 RC 振荡器）[1]
010 = $F_{OSC}/32$
001 = $F_{OSC}/8$
000 = $F_{OSC}/2$

注 1： 如果选择了 F_{RC} 时钟源，在 A/D 时钟启动之前会加上一个 T_{CY}（指令周期）的延时。这可以保证在开始转换之前执行 SLEEP 指令。

可通过软件选择将器件的正负电源电压（VDD 和 VSS）或 RA3/AN3/VREF+ 和 RA2/AN2/VREF- 引脚上的电压作为 A/D 转换的模拟参考电压。

A/D 转换器具有可在休眠状态下工作的特性。要使 A/D 转换器在休眠状态下工作，其时钟必须来自于 A/D 转换器内部的 RC 振荡器。

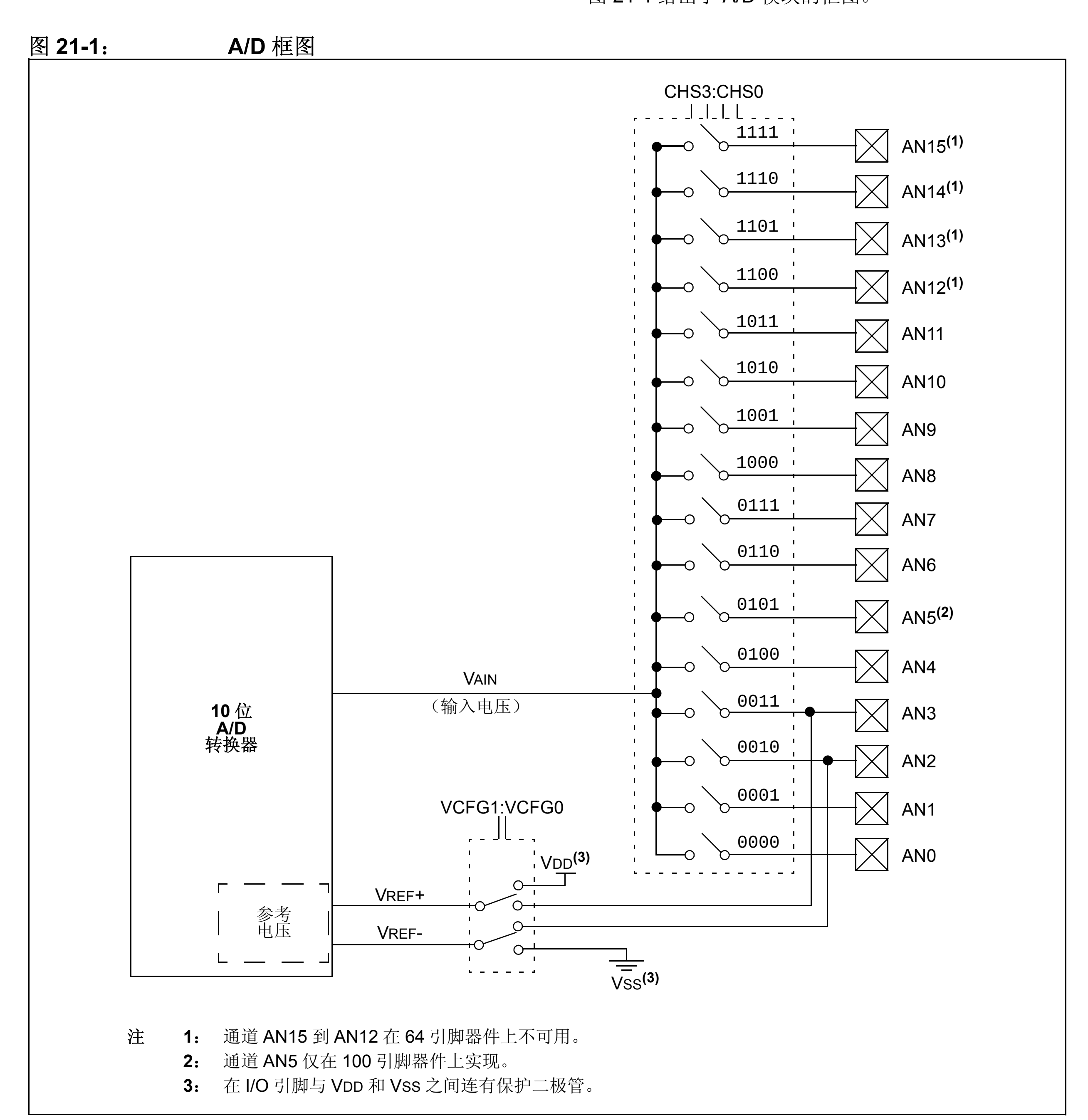
采样保持电路的输出是转换器的输入，A/D 转换器采用逐次逼近法得到转换结果。

与 A/D 转换器相关的每个端口引脚都可以被配置为模拟输入或数字 I/O。ADRESH 和 ADRESL 寄存器保存 A/D 转换的结果。当 A/D 转换完成时，结果被装入 ADRESH:ADRESL 寄存器，GO/DONE 位（ADCON0<1>）被清零且 A/D 中断标志位 ADIF 被置 1。

器件复位将强制所有寄存器进入复位状态。这将迫使 A/D 模块关闭并中止正在进行的转换。上电复位时，ADRESH:ADRESL 寄存器对的值保持不变。上电复位后，这些寄存器的值不确定。

图 21-1 给出了 A/D 模块的框图。

图 21-1： A/D 框图



注 1： 通道 AN15 到 AN12 在 64 引脚器件上不可用。
2： 通道 AN5 仅在 100 引脚器件上实现。
3： 在 I/O 引脚与 VDD 和 VSS 之间连有保护二极管。

 超前信息

按要求配置好 A/D 模块后，在开始转换之前必须采样选定的通道。模拟输入通道的相应 TRIS 位必须设置为输入。采集时间的确定，请参见**第 21.1 节 “A/D 采集要求”**。在采样完成之后，即可启动 A/D 转换。采集时间可以被编程置于 GO/$\overline{\text{DONE}}$ 位置 1 和启动转换之间。

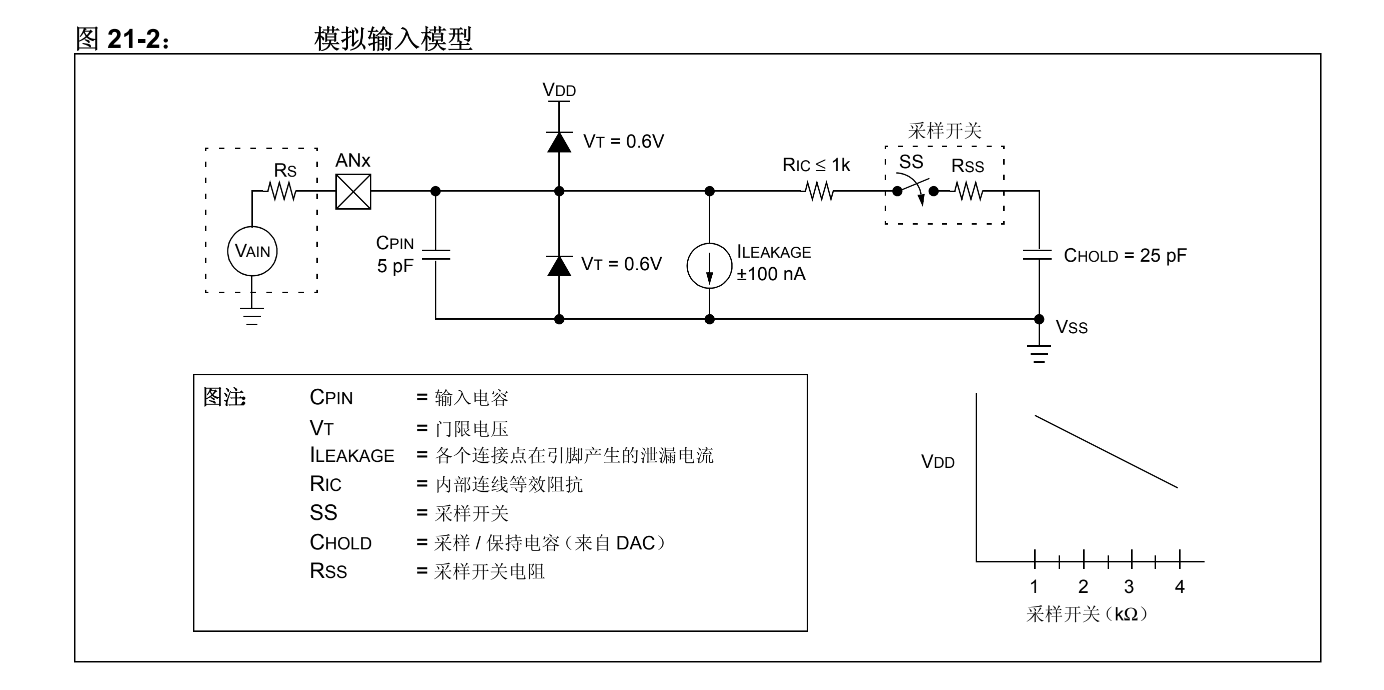
在执行 A/D 转换时应该遵循以下步骤：

1. 配置 A/D 模块：
   - 配置模拟引脚、参考电压和数字 I/O （通过 ADCON1 寄存器）
   - 选择 A/D 输入通道 （通过 ADCON0 寄存器）
   - 选择 A/D 采集时间 （通过 ADCON2 寄存器）
   - 选择 A/D 转换时钟 （通过 ADCON2 寄存器）
   - 使能 A/D 模块 （通过 ADCON0 寄存器）
2. 需要时，配置 A/D 中断：
   - ADIF 位清零
   - ADIE 位置 1
   - GIE 位置 1
3. 需要时，等待所需的采样时间。
4. 启动转换：
   - 将 GO/$\overline{\text{DONE}}$ 位 （ADCON0<1>）置 1
5. 等待 A/D 转换完成，可通过以下两种方法之一来判断转换是否完成：
   - 查询 GO/$\overline{\text{DONE}}$ 位是否被清零

   或

   - 等待 A/D 中断
6. 读取 A/D 结果寄存器（ADRESH:ADRESL），需要时将 ADIF 位清零。
7. 如需再次进行 A/D 转换，返回步骤 1 或步骤 2。每位的 A/D 转换时间定义为 T_{AD}。在下一次采样开始前需要等待至少 2T_{AD} 的时间。

图 21-2： 模拟输入模型



21.1 A/D 采集要求

为了使 A/D 转换器达到规定的精度，必须使充电保持电容（C_{HOLD}）充满至输入通道的电压电平。模拟输入模型见图 21-2。电源阻抗（R_S）和内部采样开关阻抗（R_{SS}）直接影响给电容 C_{HOLD} 充电所需要的时间。采样开关阻抗（R_{SS}）值随器件电压（V_{DD}）变化而改变。电源阻抗影响模拟输入的失调电压（由于引脚泄漏电流的原因）。**模拟信号源的最大阻抗推荐值为 2.5 kΩ**。选择（改变）模拟输入通道后，必须对通道进行采样才能启动转换，采集时间必须大于最小采集时间。

注：	当开始转换时，将保持电容与输入引脚断开。

可以使用公式 21-1 来计算最小采集时间。该公式假设误差为 1/2 LSb（A/D 转换需要 1024 步）。1/2 LSb 误差是 A/D 达到规定分辨率所允许的最大误差。

公式 21-3 显示了所需的最小采集时间 T_{ACQ} 的计算过程。计算结果基于以下假设：

C_{HOLD}	=	25 pF
Rs	=	2.5 kΩ
转换误差	≤	1/2 LSb
V_{DD}	=	3V → Rss = 2 kΩ
温度	=	85°C （系统最大值）

公式 21-1： 采集时间

$$T_{ACQ} = \text{放大器稳定时间} + \text{保持电容充电时间} + \text{温度系数}$$
$$= T_{AMP} + T_C + T_{COFF}$$

公式 21-2： A/D 最小充电时间

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
或
$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)$$

公式 21-3： 计算所需要的最小采集时间

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$

$$T_{AMP} = 0.2\ \mu s$$

$$T_{COFF} = (Temp - 25°C)(0.02\ \mu s/°C)$$
$$= (85°C - 25°C)(0.02\ \mu s/°C)$$
$$= 1.2\ \mu s$$

只有在温度 > 25°C 时才需要温度系数。当温度低于 25°C 时， $T_{COFF} = 0$ ms。

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)\ \mu s$$
$$= -(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega)\ln(0.0004883)\ \mu s$$
$$= 1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1\ \mu s + 1.2\ \mu s$$
$$= 2.4\ \mu s$$

21.2 选择和配置自动采集时间

用户可以利用 ADCON2 寄存器选择采集时间，该采集时间发生在每次 GO/DONE 位置 1 之后。

当 GO/DONE 位被置 1 时，采样停止并开始转换。用户必须确保在选择输入通道和将 GO/DONE 位置 1 之间已插入了所需的采集时间。这发生在 ACQT2:ACQT0 位（ADCON2<5:3>）保持在其 “复位” 状态 （000）的情况下，与不提供可编程采集时间的器件相兼容。

如果需要，可设置 ACQT 位以便为 A/D 模块选择可编程采集时间。当 GO/DONE 位被置 1 时， A/D 模块会继续在选定采集时间内采样输入通道，然后自动开始一次转换。由于采集时间是编程的，因此没有必要在选择通道和将 GO/DONE 位置 1 之间等待一个采集时间。

在这两种情况下，当转换完成时， GO/DONE 位均被清零， ADIF 标志位均被置 1 并且 A/D 开始再次对当前选择的通道进行采样。如果采集时间已经被编程，那么将不会有任何指示位显示采集时间是否结束，或转换是否开始。

21.3 选择 A/D 转换时钟

每位的 A/D 转换时间定义为 TAD。每完成一次 10 位 A/D 转换需要 11 个 TAD。可用软件选择 A/D 转换的时钟源。

TAD 可有以下 7 种选择：

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- 内部 RC 振荡器

为了实现正确的 A/D 转换， A/D 转换时间 （TAD）必须尽可能得小，但它必须大于最小 TAD。更多信息，请参见表 27-27 中的 A/D 参数 130 （**“A/D 转换要求”**）。

表 21-1 显示了器件在不同的工作频率下和选择不同的 A/D 时钟源时得到的 TAD。

表 21-1： 不同器件工作频率下的 TAD

A/D 时钟源 （TAD）		最高器件频率
工作状态	ADCS2:ADCS0	
2 TOSC	000	2.68 MHz
4 TOSC	100	5.71 MHz
8 TOSC	001	11.43 MHz
16 TOSC	101	22.86 MHz
32 TOSC	010	41.67 MHz
64 TOSC	110	41.67 MHz
RC(2)	x11	1.00 MHz(1)

注 1： RC 源的典型 TAD 时间为 4 ms。
2： 请参见表 27-27 中 A/D RC 时钟规范的参数 130。

21.4 配置模拟端口引脚

ADCON1、 TRISA、 TRISF 和 TRISH 寄存器控制 A/D 端口引脚的操作。如果希望端口引脚为模拟输入，则必须将相应的 TRIS 位置 1（输入）。如果将 TRIS 位清零（输出），则数字输出电平 （VOH 或 VOL）将被转换。

A/D 转换与 CHS3:CHS0 位及 TRIS 位的状态无关。

> 注 1： 读取端口寄存器时，所有配置为模拟输入通道的引脚均读为 0 （低电平）。配置为数字输入的引脚将按模拟输入进行转换。配置为数字输入的引脚将模拟输入电平精确转换为数字引脚电平。
>
> 2： 定义为数字输入引脚上的模拟电平可能会导致数字输入缓冲区消耗的电流超出器件规范。

21.5 A/D 转换

图 21-3 显示了在 GO/$\overline{DONE}$ 位置 1 且 ACQT2:ACQT0 位被清零后 A/D 转换器的工作状态。转换在下一条指令执行之后开始，以允许器件在转换开始之前进入休眠模式。

图 21-4 显示了在 GO/$\overline{DONE}$ 位置 1，ACQT2:ACQT0 位被设置为 `010`，且在转换开始之前选择 4 TAD 采集时间后 A/D 转换器的工作状态。

在转换期间将 GO/$\overline{DONE}$ 位清零将中止当前的 A/D 转换。不会用尚未完成的 A/D 转换结果更新 A/D 结果寄存器对。这意味着 ADRESH:ADRESL 寄存器仍将保持上一次转换的结果（或上一次写入 ADRESH:ADRESL 寄存器的值）。

在 A/D 转换完成或停止以后，需要等待 2 个 TAD 才能开始下一次采集。等待时间一到，将自动开始对所选通道进行采集。

注：	不应在启动A/D模块的指令中将GO/$\overline{DONE}$ 位置 1。

21.6 ECCP2 触发器的使用

ECCP2 模块的特殊事件触发信号可以启动 A/D 转换。这需要将 CCP2M3:CCP2M0 位（CCP2CON<3:0>）设置为 `1011`，且使能 A/D 模块（ADON 位置 1）。发生触发事件时，GO/$\overline{DONE}$ 位被置 1，启动 A/D 采集和转换并将 Timer1（或 Timer3）计数器复位为 0。复位 Timer1（或 Timer3）可自动重复 A/D 采集周期，最大限度地降低了软件开销（将 ADRESH:ADRESL 内容移到指定单元）。特殊事件触发信号将 GO/$\overline{DONE}$ 位置 1（启动转换）之前，用户必须选择正确的模拟输入通道和最小采集时间，或选择合适的 TACQ 时间。

如果未使能 A/D 模块（ADON 清零），则特殊事件触发信号将被 A/D 模块忽略，但它仍会将 Timer1（或 Timer3）计数器复位。

图 21-3： A/D 转换 TAD 周期 （ACQT2:ACQT0 = `000`， TACQ = `0`）

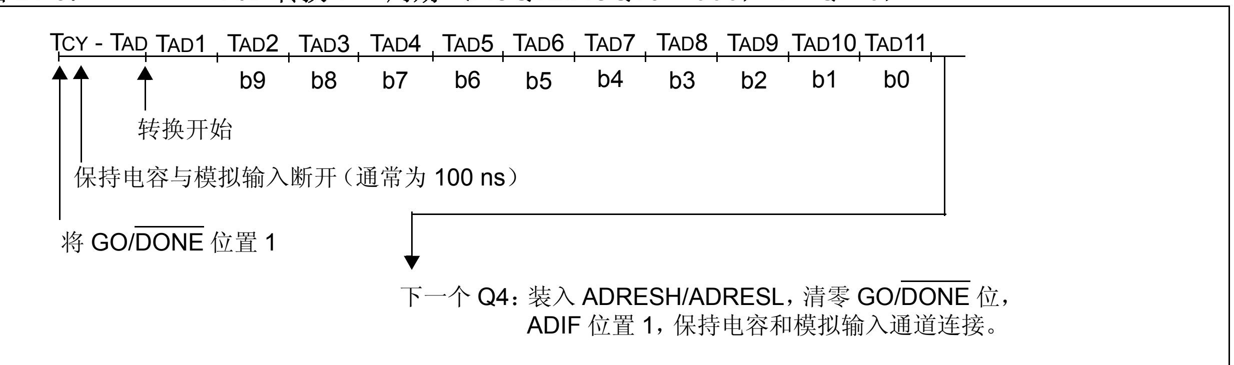
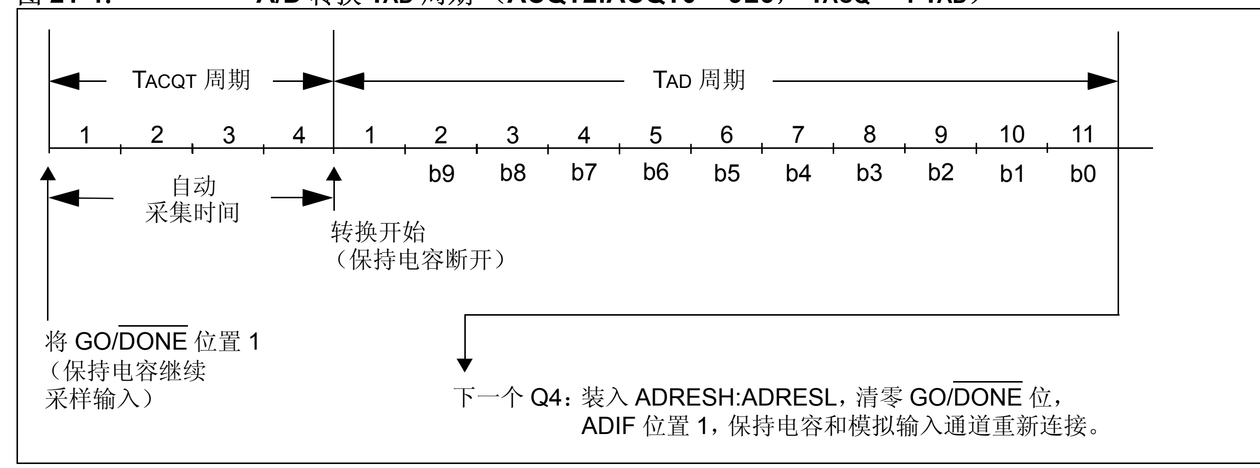


图 21-4： A/D 转换 TAD 周期 （ACQT2:ACQT0 = `010`， TACQ = 4 TAD）



21.7 A/D 转换器校准

PIC18F97J60 系列器件中的 A/D 转换器包括自校准功能，能补偿模块产生的任何偏移。校准过程是通过将 ADCAL 位（ADCON0<7>）置 1 而自动启动的。GO/DONE 位下次置 1 时，模块将执行“假”转换（即不读输入通道），将结果值内部存储起来，补偿偏移。这样就补偿了后续偏移。

校准过程假定器件处于相对稳态运行条件下。如果使用了 A/D 校准功能，应在每次器件复位后或运行条件有重大变化时执行校准操作。

21.8 在功耗管理模式下的操作

在功耗管理模式中，自动采集时间和 A/D 转换时钟的选择一定程度上可由时钟源和频率决定。

如果希望器件处于功耗管理模式时进行 A/D 采集转换，就应该根据该模式下使用的时钟对 ADCON2 中的 ACQT2:ACQT0 和 ADCS2:ADCS0 位进行更新。在进入功耗管理模式之后（两种功耗管理运行模式之一），就可以开始 A/D 采集或转换。采集或转换开始以后，器件应继续使用相同的功耗管理模式时钟源直到转换完成。如果需要，在转换期间也可以将器件置于相应的功耗管理空闲模式。

如果功耗管理模式的时钟频率小于 1 MHz，就应该选择 A/D RC 时钟源。

在休眠模式下工作需要选择 A/D RC 时钟。如果将 ACQT2:ACQT0 设置为 000 并启动 A/D 转换，转换将延时一个指令周期以允许执行SLEEP指令并进入休眠模式。OSCCON 寄存器中的 IDLEN 和 SCS 位必须在转换开始之前被清零。

表 21-2： A/D 寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	61
ADRESH	A/D 结果寄存器的高字节								60
ADRESL	A/D 结果寄存器的低字节								60
ADCON0	ADCAL	—	CHS3	CHS3	CHS1	CHS0	GO/DONE	ADON	60
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	60
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	60
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	60
PORTA	RJPU	—	RA5	RA4	RA3	RA2	RA1	RA0	62
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	61
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0(1)	62
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0(1)	61
PORTH(2)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	62
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	61

图注： — = 未用（读为 0），r = 保留。A/D 转换不使用阴影单元。

注 1： 仅在 100 引脚器件上实现。

2： 在 64 引脚器件上未实现此寄存器。

注：

22.0 比较器模块

模拟比较器模块包含两个比较器，可以用多种方式对它们进行配置。该比较器的输入可以是与 RF1 到 RF6 引脚复用的模拟输入，也可以为片上参考电压（见**第 23.0 节“比较器参考电压模块”**）。数字输出（正常或翻转的）可从引脚电平读取，也可通过控制寄存器读取。

CMCON 寄存器（寄存器 22-1）选择比较器的输入和输出配置。图 22-1 给出了各种比较器配置的框图。

寄存器 22-1： CMCON：比较器控制寄存器

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **C2OUT**：比较器 2 输出位

当 C2INV = 0 时：
1 = C2 V_{IN+} > C2 V_{IN-}
0 = C2 V_{IN+} < C2 V_{IN-}

当 C2INV = 1 时：
1 = C2 V_{IN+} < C2 V_{IN-}
0 = C2 V_{IN+} > C2 V_{IN-}

bit 6 **C1OUT**：比较器 1 输出位

当 C1INV = 0 时：
1 = C1 V_{IN+} > C1 V_{IN-}
0 = C1 V_{IN+} < C1 V_{IN-}

当 C1INV = 1 时：
1 = C1 V_{IN+} < C1 V_{IN-}
0 = C1 V_{IN+} > C1 V_{IN-}

bit 5 **C2INV**：比较器 2 输出翻转位
1 = C2 输出翻转
0 = C2 输出不翻转

bit 4 **C1INV**：比较器 1 输出翻转位
1 = C1 输出翻转
0 = C1 输出不翻转

bit 3 **CIS**：比较器输入切换位

当 CM2:CM0 = 110 时：
1 = C1 V_{IN-} 连接到 RF5/AN10/CV_{REF}
C2 V_{IN-} 连接到 RF3/AN8
0 = C1 V_{IN-} 连接到 RF6/AN11
C2 V_{IN-} 连接到 RA4/AN9

bit 2-0 **CM2:CM0**：比较器模式位

图 22-1 给出了比较器的几种模式以及相应 CM2:CM0 位的设置。

22.1 比较器配置

图 22-1 给出了比较器的 8 种工作模式。 CMCON 寄存器的 CM2:CM0 位用于选择模式。TRISF 寄存器控制每种模式下比较器引脚的数据方向。如果改变比较器模式，由于存在特定的模式改变延时 （如第 **27.0** 节 “**电气特性**” 所示），比较器的输出电平可能会在此延时期间无效。

注：	改变比较器模式期间应禁止比较器中断；否则会产生错误的中断。

图 22-1： 比较器 I/O 工作模式

禁止比较器输出
CM2:CM0 = 000
RF6/AN11 A VIN- ; RF5/AN10/CVREF A VIN+ ; C1 关（读为 0）
RF4/AN9 A VIN- ; RF3/AN8 A VIN+ ; C2 关（读为 0）

比较器关闭（POR 默认值）
CM2:CM0 = 111
RF6/AN11 D VIN- ; RF5/AN10/CVREF D VIN+ ; C1 关（读为 0）
RF4/AN9 D VIN- ; RF3/AN8 D VIN+ ; C2 关（读为 0）

两个独立的比较器
CM2:CM0 = 010
RF6/AN11 A VIN- ; RF5/AN10/CVREF A VIN+ ; C1 C1OUT
RF4/AN9 A VIN- ; RF3/AN8 A VIN+ ; C2 C2OUT

两个带输出的独立比较器
CM2:CM0 = 011
RF6/AN11 A VIN- ; RF5/AN10/CVREF A VIN+ ; C1 C1OUT
RF2/AN7/C1OUT*
RF4/AN9 A VIN- ; RF3/AN8 A VIN+ ; C2 C2OUT
RF1/AN6/C2OUT*

两个具有公共参考端的比较器
CM2:CM0 = 100
RF6/AN11 A VIN- ; RF5/AN10/CVREF A VIN+ ; C1 C1OUT
RF4/AN9 A VIN- ; RF3/AN8 D VIN+ ; C2 C2OUT

两个具有公共参考端且带输出的比较器
CM2:CM0 = 101
RF6/AN11 A VIN- ; RF5/AN10/CVREF A VIN+ ; C1 C1OUT
RF2/AN7/C1OUT*
RF4/AN9 A VIN- ; RF3/AN8 D VIN+ ; C2 C2OUT
RF1/AN6/C2OUT*

一个带输出的独立比较器
CM2:CM0 = 001
RF6/AN11 A VIN- ; RF5/AN10/CVREF A VIN+ ; C1 C1OUT
RF2/AN7/C1OUT*
RF4/AN9 D VIN- ; RF3/AN8 D VIN+ ; C2 关（读为 0）

两个比较器复用四路输入
CM2:CM0 = 110
RF6/AN11 A ; RF5/AN10/CVREF A ; CIS = 0 ; CIS = 1 ; VIN- ; VIN+ ; C1 C1OUT
RF4/AN9 A ; RF3/AN8 A ; CIS = 0 ; CIS = 1 ; VIN- ; VIN+ ; C2 C2OUT
CVREF 来自 VREF 模块

A = 模拟输入，端口始终读为 0　　D = 数字输入　　CIS（CMCON<3>）是比较器输入切换位
* 将 TRISF<2:1> 位置 1 会通过把引脚配置为输入引脚而禁止比较器输出。

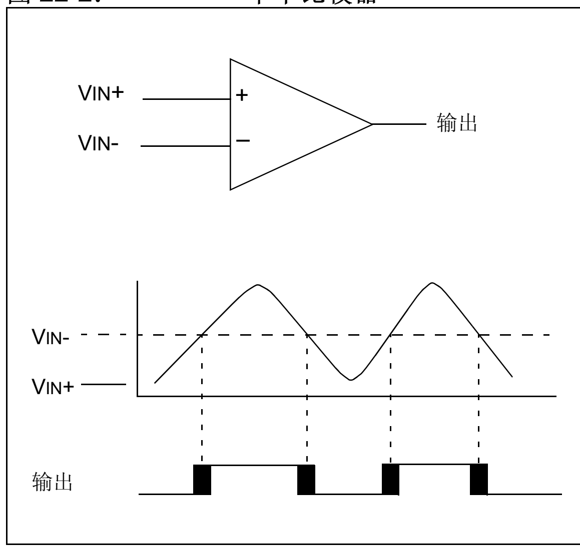
22.2 比较器工作原理

图 22-2 显示了单个比较器，以及其模拟输入电平和数字输出之间的关系。当 VIN+ 上的模拟输入电平值小于 VIN- 上的模拟输入值时，比较器输出数字低电平。当 VIN+ 上的模拟输入电平值大于 VIN- 上的模拟输入值时，比较器输出数字高电平。图 22-2 中比较器输出的阴影部分表示因输入失调电压和响应时间所造成的输出不确定区域。

22.3 比较器参考电压

根据不同的比较器工作模式，可选择使用外部或内部参考电压。将 VIN- 上的模拟信号与 VIN+ 上的信号作比较，并相应地调整比较器的数字输出 （图 22-2）。

图 22-2： 单个比较器



22.3.1 外部参考电压信号

当使用外部参考电压时，可将比较器模块中的两个比较器配置为使用同一个参考源或使用不同的参考源。但是，门限检测电路可能要求使用同一个参考源。参考信号幅值必须在 VSS 和 VDD 之间，并且可被施加到比较器的任一引脚上。

22.3.2 内部参考电压信号

比较器模块也可以选择使用内部比较器参考电压模块产生的参考电压。在第 **23.0 节 “比较器参考电压模块”** 中详细介绍了该模块。

只有在两个比较器复用四路输入的模式（CM2:CM0 = `110`）中才可使用内部参考电压。在该模式下，内部参考电压被施加到两个比较器的 VIN+ 引脚上。

22.4 比较器响应时间

响应时间是指从选定一个新的参考电压或输入源到比较器输出达到一个有效电平的最短时间。如果内部参考电压改变了，在使用比较器输出时，必须考虑内部参考电压的最大延时。否则，应使用比较器的最大延时（见第 **27.0 节 “电气特性”**）。

22.5 比较器输出

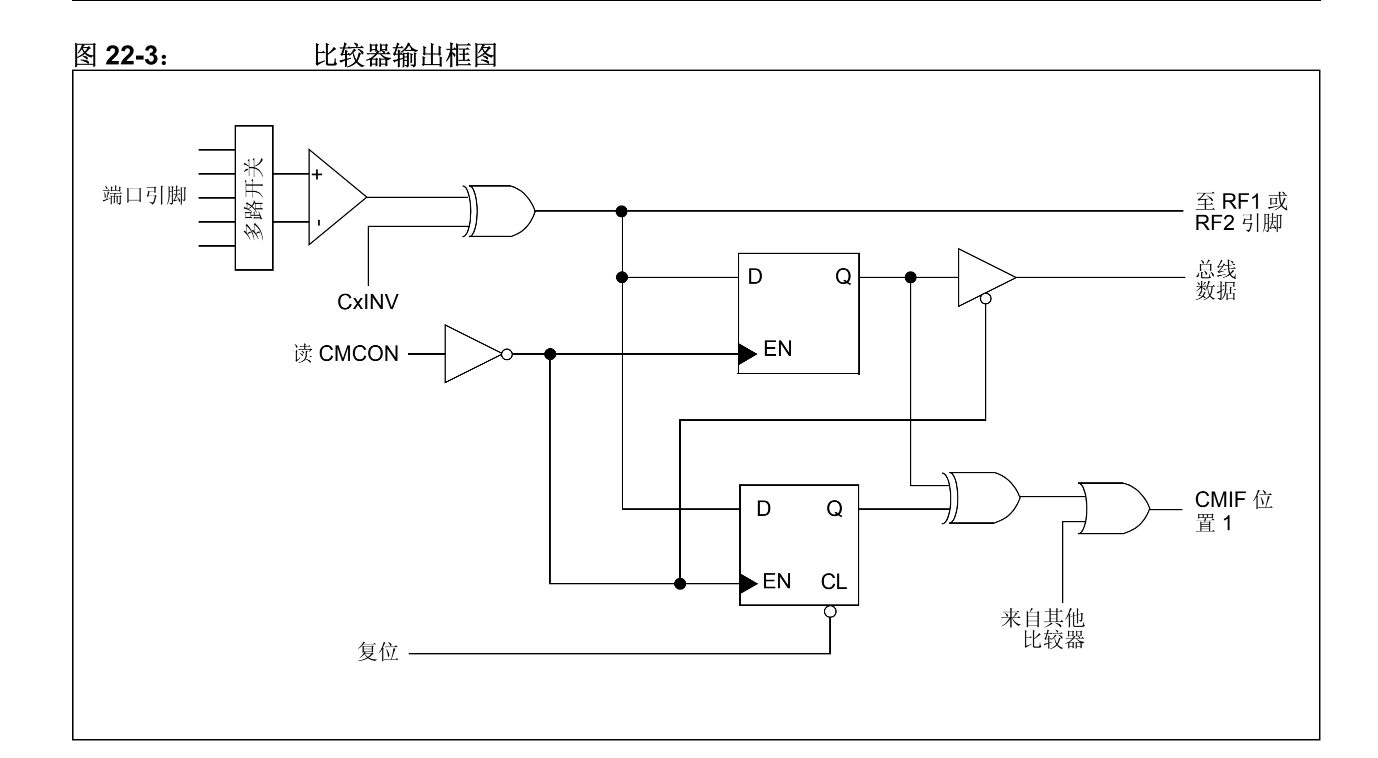
通过 CMCON 寄存器可读取比较器输出。这些位是只读的。比较器输出也可以直接输出到 RF1 和 RF2 I/O 引脚。当被使能时， RF1 和 RF2 引脚输出路径上的多路开关会发生切换，并且每个引脚输出的信号与比较器输出信号是异步的。每个比较器输出的不确定区域的大小与规范中给出的输入失调电压和响应时间有关。图 22-3 给出了比较器的输出框图。

在该模式下， TRISF 仍作为 RF1 和 RF2 引脚的输出使能 / 禁止位。

使用 C2INV 和 C1INV 位（CMCON<5:4>）可改变比较器输出的极性。

> **注 1：** 读端口寄存器时，所有配置为模拟输入的引脚将读为 `0`。配置为数字输入的引脚将根据施密特触发器输入规范，对模拟输入进行相应转换。
>
> **2：** 定义为数字输入引脚上的模拟电平可能会使输入缓冲区的电流消耗超过规定值。

图 22-3： 比较器输出框图



22.6 比较器中断

任一比较器的输出值发生变化，都会将该比较器的中断标志位置 1。需要用软件保存输出位的状态信息 （从 CMCON<7:6> 读取），以确定实际发生的变化。 CMIF 位 （PIR2<6>）是比较器中断标志位，且必须通过清零复位。由于可以向该寄存器写入 1，因此可以产生模拟中断。

必须将 CMIE 位（PIE2<6>）和 PEIE 位（INTCON<6>）置1以允许中断。此外，还必须将GIE位（INTCON<7>）置 1。如果这些位中的任何一个被清零，将无法允许中断，尽管中断条件发生时仍会将 CMIF 位置 1。

注：	当执行读操作时 （Q2 周期开始），如果 CMCON 寄存器 （C1OUT 或 C2OUT）的值发生变化，那么 CMIF （PIR2 寄存器）中断标志位可能不会被置 1。

用户可用以下方式在中断服务程序中清除该中断：

a） 任何对 CMCON 的读或写都将结束不匹配条件。

b） 将标志位 CMIF 清零。

不匹配条件将继续把标志位 CMIF 置 1。读 CMCON 将结束不匹配条件并允许将标志位 CMIF 清零。

22.7 休眠期间的比较器操作

当比较器处于激活状态而器件处于休眠模式时，比较器仍保持激活状态并可产生中断 （如果中断被允许）。中断会把器件从休眠模式唤醒。每个处于工作状态中的比较器都会消耗额外的电流，如比较器规范中所示。若要将休眠模式下的功耗减到最低，可在进入休眠模式前关闭比较器（CM2:CM0 = 111）。如果器件从休眠状态唤醒， CMCON 寄存器的内容不受影响。

22.8 复位的影响

器件复位强制 CMCON 寄存器进入复位状态，从而使比较器模块进入关闭模式（CM2:CM0 = 111）。但是，器件复位时输入引脚 （RF3 到 RF6）被默认配置为模拟输入。这些引脚的 I/O 配置由 PCFG3:PCFG0 位（ADCON1<3:0>）的设置决定。因此，当复位时引脚呈现模拟输入状态，将使得器件电流降至最小。

 超前信息

22.9 模拟输入连接注意事项

模拟输入的简化电路如图 22-4 所示。由于模拟引脚被连接到数字输出端，它们与 VDD 和 VSS 之间连有反向偏置的二极管。因此，模拟输入必须在 VSS 和 VDD 之间。如果输入电压超出该范围 0.6V 以上，就可能发生一个二极管正向偏置而使输入电压箝位。模拟信号源的最大阻抗推荐值为 10 kΩ。任何连接到模拟输入引脚的外部元件（如电容或齐纳二极管），要保证其泄漏电流极小。

图 22-4： 比较器模拟输入模型

表 22-1： 与比较器模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	61
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	60
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	60
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	62
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	61

图注： — = 未用 （读为 0）， r = 保留。比较器模块不使用阴影单元。

注：

23.0 比较器参考电压模块

比较器参考电压模块是一个 16 阶的梯形电阻网络，可提供多个参考电压供选择。虽然它的主要目的是为模拟比较器提供参考电压，但也可将它用于其他场合。

图 23-1 给出了此模块的框图。梯形电阻经过分段可提供两种范围的 CVREF 值，并且该网络还具有断电功能，可以在不使用参考电压的情况下节省功耗。器件的 VDD/VSS 或外部参考电压都可以作为此模块的参考电源。

23.1 配置比较器参考电压

参考电压模块是通过 CVRCON 寄存器（寄存器 23-1）来控制的。比较器参考电压模块提供两种范围的输出电压，每种范围都具有 16 个不同的电平。CVRR 位（CVRCON<5>）选择输出电压的范围。这两种范围的主要区别在于其电压值之间的步长不同（其中一种范围可提供较高的分辨率），该步长由 CVREF 选择位（CVR3:CVR0）来决定。下面是计算比较器参考电压输出值的公式：

如果 CVRR = 1：
CVREF = ((CVR3:CVR0)/24) x (CVRSRC)

如果 CVRR = 0：
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) x (CVRSRC)

比较器参考电压模块的电源可以来自 VDD 和 VSS，也可以来自与 RA2 和 RA3 复用的外部 VREF+ 和 VREF-。电压源由 CVRSS 位（CVRCON<4>）选择。

在更改 CVREF 输出值时，必须考虑比较器参考电压的稳定时间（见**第 27.0 节 "电气特性"**中的表 27-3）。

寄存器 23-1： CVRCON：比较器参考电压控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE(1)	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7 **CVREN**：比较器参考电压使能位
1 = CVREF 电路上电
0 = CVREF 电路断电

bit 6 **CVROE**：比较器 VREF 输出使能位 (1)
1 = CVREF 电平也从 RF5/AN10/CVREF 引脚输出
0 = CVREF 电压与 RF5/AN10/CVREF 引脚断开

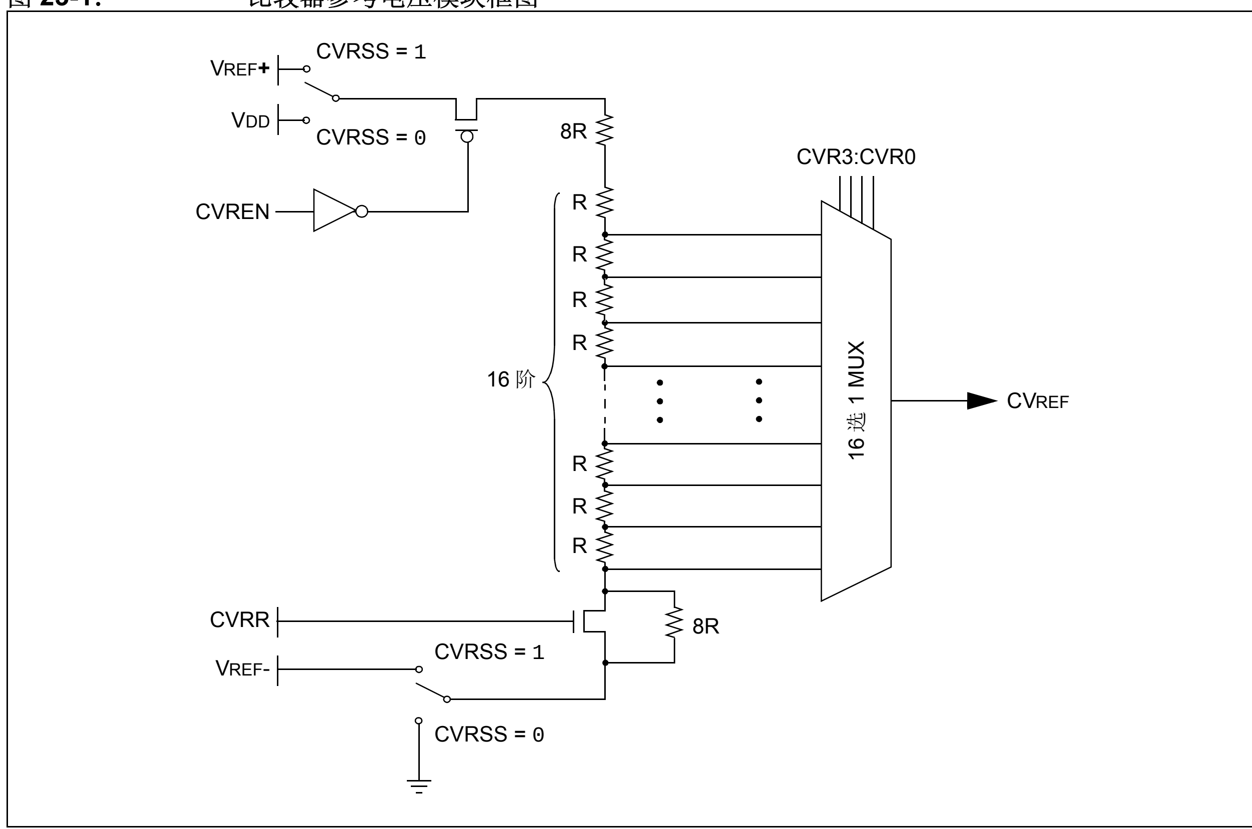
bit 5 **CVRR**：比较器 VREF 范围选择位
1 = 0 到 0.667 CVRSRC，步长为 CVRSRC/24 （低电平范围）
0 = 0.25 CVRSRC 到 0.75 CVRSRC，步长为 CVRSRC/32 （高电平范围）

bit 4 **CVRSS**：比较器 VREF 源选择位
1 = 比较器参考电压源， CVRSRC = (VREF+) – (VREF-)
0 = 比较器参考电压源， CVRSRC = VDD – VSS

bit 3-0 **CVR3:CVR0**：比较器 VREF 值选择位 （0 ≤ (CVR3:CVR0) ≤ 15）
当 CVRR = 1 时：
CVREF = ((CVR3:CVR0)/24) • (CVRSRC)
当 CVRR = 0 时：
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) • (CVRSRC)

注 1： CVROE 改写 TRISF<5> 位设置。


图 23-1： 比较器参考电压模块框图



23.2 参考电压精度 / 误差

由于模块结构的限制，并不能实现整个参考电压范围的满量程输出。梯形电阻网络顶部和底部的晶体管（图 23-1）使 CVREF 值不能达到参考电压源的满幅值。参考电压是由参考电压源分压而来的，因此 CVREF 输出随参考电压源的波动而变化。经过测试的参考电压的绝对精度，请参见**第 27.0节“电气特性”**。

23.3 休眠期间的操作

如果因中断或看门狗定时器超时将器件从休眠模式唤醒， CVRCON 寄存器的内容将不受影响。为了降低休眠模式下的电流消耗，应禁止参考电压模块。

23.4 复位的影响

器件复位时， CVREN 位（CVRCON<7>）将被清零从而禁止参考电压模块。复位还将 CVROE 位（CVRCON<6>）清零，使参考电压与 RA2 引脚断开；同时通过将 CVRR 位 （CVRCON<5>）清零选择高电压范围。 CVR 值选择位也将清零。

23.5 连接注意事项

参考电压模块的工作独立于比较器模块。如果 CVROE 位被置 1，那么参考电压发生器的输出可能与 RF5 引脚相连。当 RA2 被配置为数字输入引脚时，将参考电压输出连接到 RA2 引脚，将会增加电流消耗。使能 CVRSS 时，将 RF5 用作数字输出引脚也将增加电流消耗。

RF5 引脚可被用作简单的 D/A 输出，但是其驱动能力有限。要提高电流驱动能力，VREF 参考电压输出端必须外接缓冲区。图 23-2 举例说明了这一缓冲技术。

 超前信息

图 23-2： 参考电压输出缓冲示例

注 1： R 的值取决于参考电压配置位 CVRCON<5> 和 CVRCON<3:0>。

表 23-1： 与比较器参考电压模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	60
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	60
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	61

图注： — = 未用，读为 0。比较器参考电压模块不使用阴影单元。

注：

24.0 CPU 的特殊功能

PIC18F97J60 系列器件具有几项特殊的功能旨在最大限度地提高系统可靠性，并通过减少外部元件将成本降至最低。这些功能包括：

- 振荡器选择
- 复位：
  - 上电复位 （POR）
  - 上电延时定时器 （PWRT）
  - 振荡器起振定时器 （OST）
  - 欠压复位 （BOR）
- 中断
- 看门狗定时器 （WDT）
- 故障保护时钟监视器
- 双速启动
- 代码保护
- 在线串行编程

要根据具体应用对频率、功耗、精度和成本的要求来选择振荡器。在**第 2.0 节 "振荡器配置"**中详细讨论了所有的选项。

在本数据手册的前面几章中已完整地讨论了器件的复位和中断。

PIC18F97J60 系列器件除了为复位提供上电延时定时器和振荡器起振定时器之外，还具有一个可配置的看门狗定时器，该定时器由软件控制。

器件自带的内部 RC 振荡器还提供了故障保护时钟监视器（FSCM）和双速启动这两个额外的功能。FSCM 对外设时钟进行后台监视，并在外设时钟发生故障时自动切换时钟源。双速启动使得几乎可在启动发生那一刻立即执行代码，同时主时钟源继续其起振延时。

通过设置相应的配置寄存器位可以使能和配置所有这些功能。

24.1 配置位

可以通过对配置位编程 （读为 `0`）或不编程 （读为 `1`）来选择不同的器件配置。这些配置位被映射到程序存储器以 300000h 开始的单元中。表 24-1 列出了所有的配置位。从寄存器 24-1 到寄存器 24-8 详细解释了各配置位的不同功能。

24.1.1 配置 PIC18F97J60 系列器件的注意事项

PIC18F97J60 系列器件不再使用耐久性存储寄存器存储配置信息。配置字节以易失性存储方式实现，这就意味着在器件每次上电时都必须对配置数据进行编程。

配置数据存储在片上程序存储空间顶部的 4 个字中，这些字被称为闪存配置字，如表 5-1 中所示。配置字按表 24-1 中相同的次序存储在程序存储器中，CONFIG1L 位于地址最低的单元，CONFIG3H 位于地址最高的单元。在器件上电时这些数据被自动装入正确的配置寄存器。

当为这些器件创建应用程序时，用户应该为配置数据特别分配闪存配置字单元，以确保当编译代码时程序代码不会存储在该地址上。

在上电复位时存储配置位的易失性存储单元始终复位为 `1`。对于其他类型的复位事件，将保留和使用先前已编程的值，而无需从程序存储器重新装入数据。

程序存储器中 CONFIG1H、CONFIG2H 和 CONFIG3H 的高 4 位也应为 `1111`。这样当这些配置字被意外执行到时，被当作一条 `NOP` 指令。由于配置位并未在对应的单元中真正实现，因此向这些单元写入 `1` 不会影响器件工作。

为了避免在代码执行期间配置被意外更改，可编程配置位只可被写入一次。在上电周期内对这些位进行初始化之后就不能再改写了。要改变器件的配置需要对器件重新上电。

表 24-1： 配置位和器件 ID

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	默认 / 未编程值 (1)
300000h	CONFIG1L	$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN	111- ---1
300001h	CONFIG1H	—(2)	—(2)	—(2)	—(2)	—(3)	CP0	—	—	---- 01--
300002h	CONFIG2L	IESO	FCMEN	—	—	—	FOSC2	FOSC1	FOSC0	11-- -111
300003h	CONFIG2H	—(2)	—(2)	—(2)	—(2)	WDTPS3	WDTPS2	WDTPS1	WDTPS0	---- 1111
300004h	CONFIG3L	WAIT(4)	BW(4)	EMB1(4)	EMB0(4)	EASHFT(4)	—	—	—	1111 1---
300005h	CONFIG3H	—(2)	—(2)	—(2)	—(2)	—	ETHLED	ECCPMX(5)	CCP2MX(5)	---- -111
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx(6)
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	xxxx xxxx(6)

图注： x = 未知， u = 不变， - = 未用。未使用阴影单元，读为 0。

注 1： 这些值反映出厂时和上电复位后的未编程状态。在所有其他复位状态中，配置字节保持原先的编程状态。

2： 程序存储器中这些位的值应始终为 1。这样可确保如果意外地执行了这些单元，将会执行 NOP 指令。

3： 该位应始终保持为 0。

4： 仅在 100 引脚器件上实现。

5： 仅在 80 引脚和 100 引脚器件上实现。

6： 请参见寄存器 24-7 和寄存器 24-8 查询 DEVID 的值。这些寄存器为只读寄存器，用户不能对其进行编程。

 超前信息

寄存器 24-1： CONFIG1L：配置寄存器 1 的低字节 （字节地址为 300000h）

R/WO-1	R/WO-1	R/WO-0	U-0	U-0	U-0	U-0	R/WO-1
$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7 **$\overline{\text{DEBUG}}$**：后台调试器使能位
1 = 禁止后台调试器， RB6 和 RB7 被配置为通用 I/O 引脚
0 = 使能后台调试器， RB6 和 RB7 专用于在线调试

bit 6 **XINST**：扩展指令集使能位
1 = 使能指令集扩展和变址寻址模式
0 = 禁止指令集扩展和变址寻址模式 （传统模式）

bit 5 **STVREN**：堆栈上溢 / 下溢复位使能位
1 = 使能堆栈上溢 / 下溢复位
0 = 禁止堆栈上溢 / 下溢复位

bit 4-1 未用：读为 0

bit 0 **WDTEN**：看门狗定时器使能位
1 = 使能 WDT
0 = 禁止 WDT （由 SWDTEN 位控制）

寄存器 24-2： CONFIG1H：配置寄存器 1 的高字节 （字节地址为 300001h）

U-0	U-0	U-0	U-0	U-0[1]	R/WO-1	U-0	U-0
—[2]	—[2]	—[2]	—[2]	—	CP0	—	—
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7-3 未用：读为 0

bit 2 **CP0**：代码保护位
1 = 程序存储器未受代码保护
0 = 程序存储器受代码保护

bit 1-0 未用：读为 0

注 **1**： 该位应始终保持为 0。
2： 程序存储器中这些位的值应始终为 1。这样可确保如果意外地执行了这些单元，将会执行 NOP 指令。

寄存器 24-3： **CONFIG2L：配置寄存器 2 的低字节 （字节地址为 300002h）**

R/WO-1	R/WO-1	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1
IESO	FCMEN	—	—	—	FOSC2	FOSC1	FOSC0
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7 **IESO：**双速启动 （内部 / 外部振荡器切换）控制位
1 = 使能双速启动
0 = 禁止双速启动

bit 6 **FCMEN：**故障保护时钟监视器使能位
1 = 使能故障保护时钟监视器
0 = 禁止故障保护时钟监视器

bit 5-3 未用：读为 0

bit 2 **FOSC2：**默认 / 复位系统时钟选择位
1 = 当 OSCCON<1:0> = 00 时，使用由 FOSC1:FOSC0 选择的时钟作为系统时钟
0 = 当 OSCCON<1:0> = 00 时，使用 INTRC 作为系统时钟

bit 1-0 **FOSC1:FOSC0：**振荡器选择位
11 = EC 振荡器， PLL 使能并由软件控制， OSC2 用作 CLKO 功能
10 = EC 振荡器， OSC2 用作 CLKO 功能
01 = HS 振荡器， PLL 使能并由软件控制
00 = HS 振荡器

 超前信息

寄存器 24-4： CONFIG2H：配置寄存器 2 的高字节 （字节地址为 300003h）

U-0	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—[(1)]	—[(1)]	—[(1)]	—[(1)]	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7-4　未用：读为 0

bit 3-0　**WDTPS3:WDTPS0：**看门狗定时器后分频比选择位

1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

注 1： 程序存储器中这些位的值应始终为 1。这样可确保如果意外地执行了这些单元，将会执行 NOP 指令。

寄存器 24-5： CONFIG3L：配置寄存器 3 的低字节 （字节地址为 300004h）

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT(1)	BW(1)	EMB1(1)	EMB0(1)	EASHFT(1)	—	—	—
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7 **WAIT**：外部总线等待使能位 (1)
1 = 禁止为外部存储器总线上的操作插入等待状态
0 = 使能为外部存储器总线上的操作插入等待状态，并由 MEMCON<5:4> 选择

bit 6 **BW**：数据总线宽度选择位 (1)
1 = 16 位数据宽度模式
0 = 8 位数据宽度模式

bit 5-4 **EMB1:EMB0**：外部存储器总线配置位 (1)
11 = 单片机模式，禁止外部总线
10 = 扩展单片机模式， 12 位地址模式
01 = 扩展单片机模式， 16 位地址模式
00 = 扩展单片机模式， 20 位地址模式

bit 3 **EASHFT**：外部地址总线平移使能位 (1)
1 = 使能地址平移；外部总线上的地址是以 000000h 为起始地址的偏移值
0 = 禁止地址平移；外部总线上的地址反映 PC 值

bit 2-0 未用：读为 0

注 **1**： 仅在 100 引脚器件上实现。

寄存器 24-6： CONFIG3H：配置寄存器 3 的高字节 （字节地址为 300005h）

U-0	U-0	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1
—(1)	—(1)	—(1)	—(1)	—	ETHLED	ECCPMX(2)	CCP2MX(2)
bit 7							bit 0

图注：			
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0	
-n = 未对器件编程时的值		1 = 置 1	0 = 清零

bit 7-3 未用：读为 0

bit 2 **ETHLED**：以太网 LED 使能位
1 = 当使能以太网模块时，RA0/RA1 与 LEDA/LEDB 复用；当禁止以太网模块时，RA0/RA1 用作 I/O
0 = RA0/RA1 用作 I/O，与以太网模块状态无关

bit 1 **ECCPMX**：ECCP 多路复用位 (2)
1 = ECCP1 输出 （P1B/P1C）与 RE6 和 RE5 复用；
ECCP3 输出 （P3B/P3C）与 RE4 和 RE3 复用
0 = ECCP1 输出 （P1B/P1C）与 RH7 和 RH6 复用；
ECCP3 输出 （P3B/P3C）与 RH5 和 RH4 复用

bit 0 **CCP2MX**：ECCP2 多路复用位 (2))
1 = ECCP2/P2A 与 RC1 复用
0 = 在单片机模式下 ECCP2/P2A 与 RE7 复用 （80 引脚和 100 引脚器件）
或在扩展单片机模式下与 RB3 复用 （仅 100 引脚器件）

注 **1**： 程序存储器中这些位的值应始终为 1。这样可确保如果意外地执行了这些单元，将会执行 NOP 指令。
2： 仅在 80 引脚和 100 引脚器件上实现。

寄存器 24-7： DEVID1：PIC18F97J60 系列器件的器件 ID 寄存器 1

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

图注：		
R = 只读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

bit 7-5 **DEV2:DEV0**：器件 ID 位
请参见寄存器 24-8 获取完整列表。

bit 4-0 **REV4:REV0**：版本 ID 位
这些位用于表明器件版本。

寄存器 24-8： DEVID2：PIC18F97J60 系列器件的器件 ID 寄存器 2

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

图注：		
R = 只读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

bit 7-0 **DEV10:DEV3**：器件 ID 位：

DEV10:DEV3 （DEVID2<7:0>）	DEV2:DEV0 （DEVID1<7:5>）	器件
0001 1000	000	PIC18F66J60
0001 1111	000	PIC18F66J65
0001 1111	001	PIC18F67J60
0001 1000	001	PIC18F86J60
0001 1111	010	PIC18F86J65
0001 1111	011	PIC18F87J60
0001 1000	010	PIC18F96J60
0001 1111	100	PIC18F96J65
0001 1111	101	PIC18F97J60

24.2 看门狗定时器 （WDT）

PIC18F97J60 系列器件的 WDT 是由 INTRC 振荡器驱动的。当使能 WDT 时，时钟源也将同时使能。WDT 定时周期的标称值为 4 ms，其稳定性与 INTRC 振荡器相同。

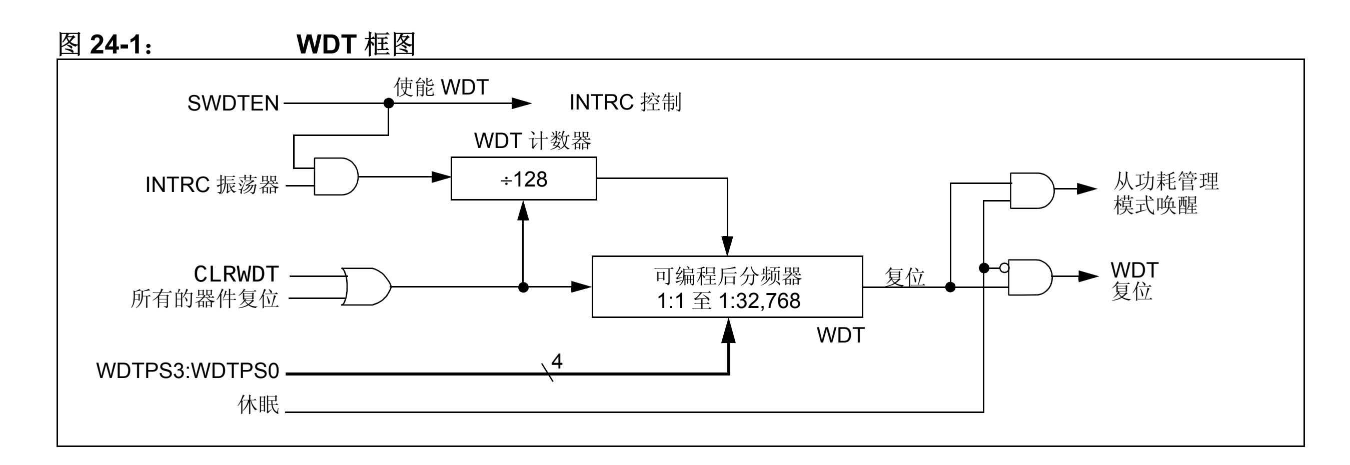
4 ms 的 WDT 定时周期将与 16 位后分频器的值相乘来得到更长的时间周期。通过配置寄存器 2H 中的 WDTPS 位来控制一个多路开关以对 WDT 后分频器的输出进行选择。因此可获得的定时周期范围为 4 ms 至 131.072 秒（2.18 分钟）。当发生以下任一事件时，WDT 和后分频器将被清零，这些事件包括：执行了 SLEEP 或 CLRWDT 指令，或者发生了时钟故障 （主时钟或 Timer1 振荡器）。

> 注 1： 当执行 CLRWDT 和 SLEEP 指令时， WDT 和后分频器的计数值将被清零。
>
> 2： 当执行 CLRWDT 指令时，后分频器的计数值将被清零。

24.2.1 控制寄存器

WDTCON 寄存器 （寄存器 24-9）是可读写寄存器。SWDTEN 位使能或禁止 WDT 操作。仅当 WDT 被配置位禁止时，才允许使用软件改写 WDTEN 配置位并使能 WDT。

图 24-1： WDT 框图



寄存器 24-9： WDTCON：看门狗定时器控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	SWDTEN(1)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = POR 值	1 = 置 1	0 = 清零	x = 未知

bit 7-1 未用：读为 0

bit 0 **SWDTEN：** 由软件控制的看门狗定时器使能位 (1)
1 = 打开看门狗定时器
0 = 关闭看门狗定时器

注 1： 当使能 WDTEN 配置位时该位不起作用。

表 24-2： 看门狗定时器寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
WDTCON	—	—	—	—	—	—	—	SWDTEN	60

图注： — = 未用，读为 0。看门狗定时器不使用阴影单元。

24.3 片上稳压器

所有的 PIC18F97J60 系列器件使用标称值为 2.5V 的电压为其内核数字逻辑供电。对于需要工作在更高电压（如典型电压值为 3.3V）下的应用，PIC18F97J60 系列的所有器件均包含一个片上稳压器，可使器件内核逻辑运行在 VDD 下。

ENVREG 引脚控制该稳压器。把 VDD 连到该引脚将使能稳压器，然后稳压后的电压通过其他VDD 引脚向内核供电。当使能稳压器时，必须将一个低 ESR 滤波电容连接到 VDDCORE/VCAP 引脚 （图 24-2），这有利于保持稳压器的稳定性。**第 27.3 节 "直流特性：PIC18F97J60 系列 （工业级）"** 中提供了该滤波电容的推荐值。

如果 ENVREG 与 VSS 相连，则禁止稳压器。在这种情况下，必须使用独立的 2.5V 标称值的电源通过 VDDCORE/VCAP 引脚为器件的内核逻辑供电，从而将 I/O 引脚驱动为一个较高的电平，通常为 3.3V。另外，VDDCORE/VCAP 和 VDD 引脚可以连在一起，使器件工作在较低的标称电压下。请参见图 24-2 了解可能的配置。

24.3.1 片上稳压器和 BOR

当使能片上稳压器时，PIC18F97J60 系列器件也会有一个简单的欠压保护功能。如果向稳压器提供的电压不足以维持一个稳定的电平，那么稳压器复位电路将产生欠压复位。 BOR 标志位 （RCON<0>）会捕捉该事件。

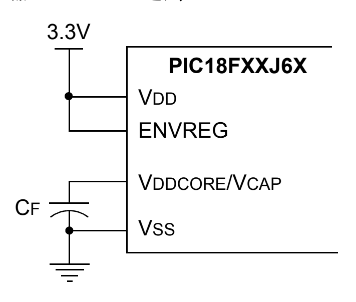
第 4.4 节 "欠压复位 （BOR）" 和 **第 4.4.1 节 "检测 BOR"** 详细描述了 BOR 的工作原理。**第 27.1 节 "直流特性：供电电压，PIC18F97J60 系列（工业级）"** 中指定了欠压复位电压值。

24.3.2 上电要求

片上稳压器是为了满足器件的上电要求而设计的。如果应用不使用稳压器，那就必须严格遵守上电条件。在上电时， VDDCORE 决不能比 VDD 高出 0.3V 以上。

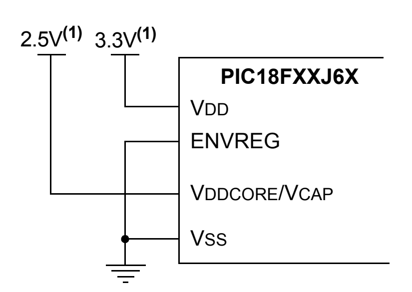
图 24-2： 片上稳压器连接

使能稳压器（**ENVREG 连到 VDD**）：

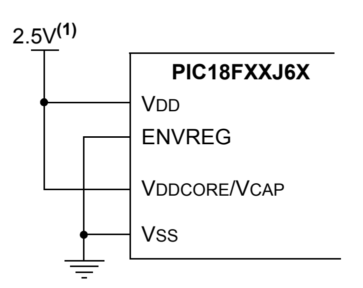


禁止稳压器（**ENVREG 接地**）：

（VDD > VDDCORE）



（VDD = VDDCORE）



注 1： 这些为典型的工作电压。请参见 **第 27.1 节 "直流特性：供电电压"** 了解 VDD 和 VDDCORE 的全部工作电压范围。

 超前信息

24.4 双速启动

双速启动功能允许单片机在主时钟源稳定之前使用 INTRC 振荡器作为时钟源，从而帮助器件最大限度地缩短从振荡器起振到代码执行之间的延时。通过将 IESO 配置位置 1 可使能该功能。

仅当主振荡器模式为 HS 或 HSPLL （基于晶振）时才可使用双速启动。由于 EC 和 ECPLL 模式不需要 OST 起振延时，因此应禁止双速启动。

一旦使能双速启动，当器件复位或从休眠模式中被唤醒时，在上电延时定时器发生超时后，器件将被配置成使用内部振荡电路作为时钟源。这使得在主振荡器起振、OST 运行的同时，代码开始执行。一旦 OST 超时，器件就自动切换到 PRI_RUN 模式。

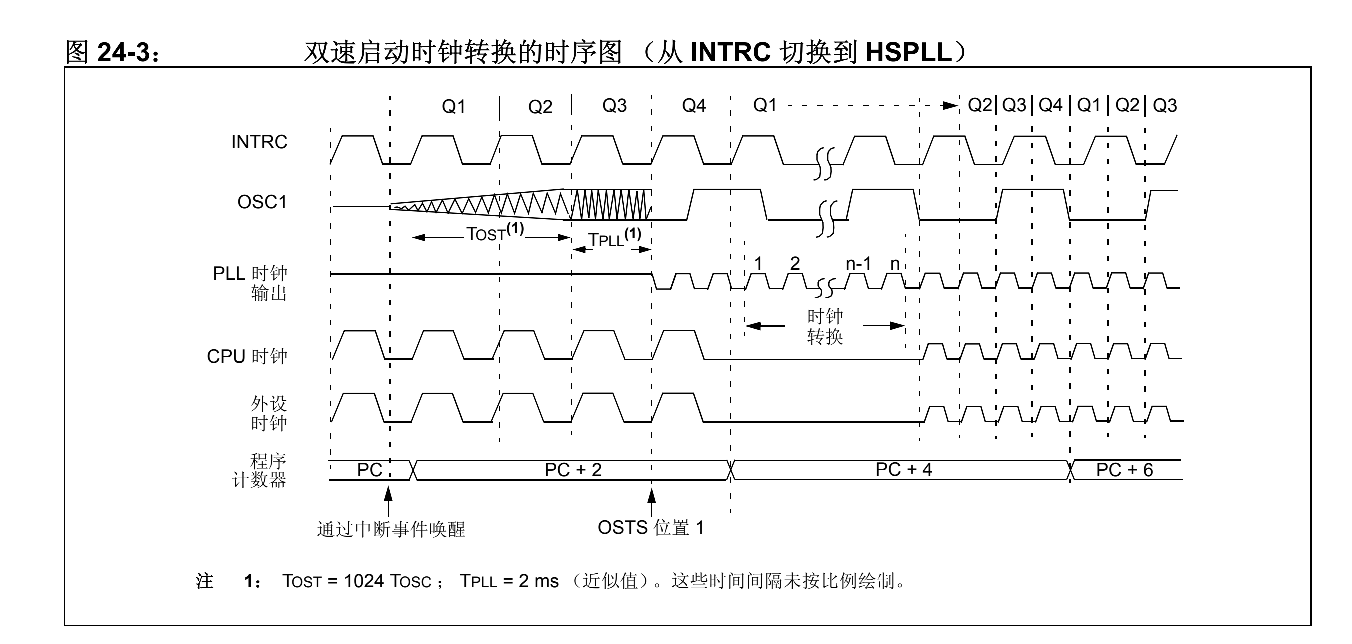
在其他功耗管理模式下，不使用双速启动。器件将使用当前选定的时钟源直到主时钟源可用为止。 IESO 位的设置被忽略。

24.4.1 使用双速启动时的注意事项

当在双速启动模式中使用 INTRC 振荡器时，器件仍将遵守进入功耗管理模式的正常指令顺序，包括执行多条 SLEEP 指令 （见**第 3.1.4 节 “多条 SLEEP 命令”**）。实际上，这意味着在 OST 超时前用户代码可以改变 SCS1:SCS0 位的设置或执行 SLEEP 指令。这就使应用程序能短暂地唤醒器件，执行 “日常事务”，并在器件开始使用主时钟源前返回休眠状态。

用户代码还能通过检查 OSTS 位 （OSCCON<3>）的状态来确定主时钟源是否正在为系统提供时钟。如果该位置 1，则表示主振荡器正在为系统提供时钟。否则，表示当器件复位或从休眠模式被唤醒期间由内部振荡器电路为系统提供时钟。

图 24-3： 双速启动时钟转换的时序图 （从 **INTRC** 切换到 **HSPLL**）



注 1： T_{OST} = 1024 T_{OSC}； T_{PLL} = 2 ms （近似值）。这些时间间隔未按比例绘制。

24.5 故障保护时钟监视器

故障保护时钟监视器（FSCM）可使单片机在外部时钟发生故障时，自动将系统时钟切换到内部振荡器电路以保证器件能继续运行。将 FCMEN 配置位置 1 可使能 FSCM 功能。

当使能 FSCM 时， INTRC 振荡器将一直保持运行以监视外设时钟，并且在外设时钟发生故障时作为备用时钟。时钟监视 （如图 24-4 所示）通过创建一个采样时钟信号实现，该信号为 INTRC 输出的 64 分频。这样就使得 FSCM 采样时钟脉冲之间有充足的时间间隔，从而保证在此期间至少有一个外设时钟沿出现。外设时钟和采样时钟作为时钟监视锁存器 （CM）的输入。 CM 在器件时钟源的下降沿被置 1，在采样时钟的上升沿被清零。

图 24-4： FSCM 框图

时钟监视器
锁存器（CM）
（边沿触发）
外设时钟
S Q
INTRC 源
(32 µs)
÷ 64
488 Hz
（2.048 ms）
C Q
检测到
时钟故障

在采样时钟的下降沿检测外部时钟故障。如果在出现采样时钟的下降沿时， CM 仍置 1，就表示检测到外部时钟故障 （图 24-5）。这将引发以下事件：

- 通过将 OSCFIF（PIR2<7>）置 1，由 FSCM 产生振荡器故障中断；
- 器件时钟源切换为内部振荡器电路 （OSCCON 不会被更新，因此无法显示当前时钟源——这就是故障保护状态）；并且
- WDT 复位。

切换过程中，对于时序要求较高的应用，内部振荡器电路的后分频频率可能不够稳定。在这些情况下，最好选择另一种时钟配置并进入其他功耗管理模式。可以尝试部分恢复或执行安全的关闭。更多详细信息，请参见**第 3.1.4 节“多条 SLEEP 命令”**和**第 24.4.1 节“使用双速启动时的注意事项”**。

FSCM 只能检测出主时钟源或辅助时钟源的故障。如果内部振荡器电路发生故障，将不会被检测到，当然也不可能采取任何措施。

24.5.1 FSCM 和看门狗定时器

FSCM 和 WDT 均以 INTRC 振荡器作为时钟源。由于 WDT 使用独立的分频器和计数器，使能 FSCM 时，禁止 WDT 对 INTRC 振荡器的运行没有任何影响。

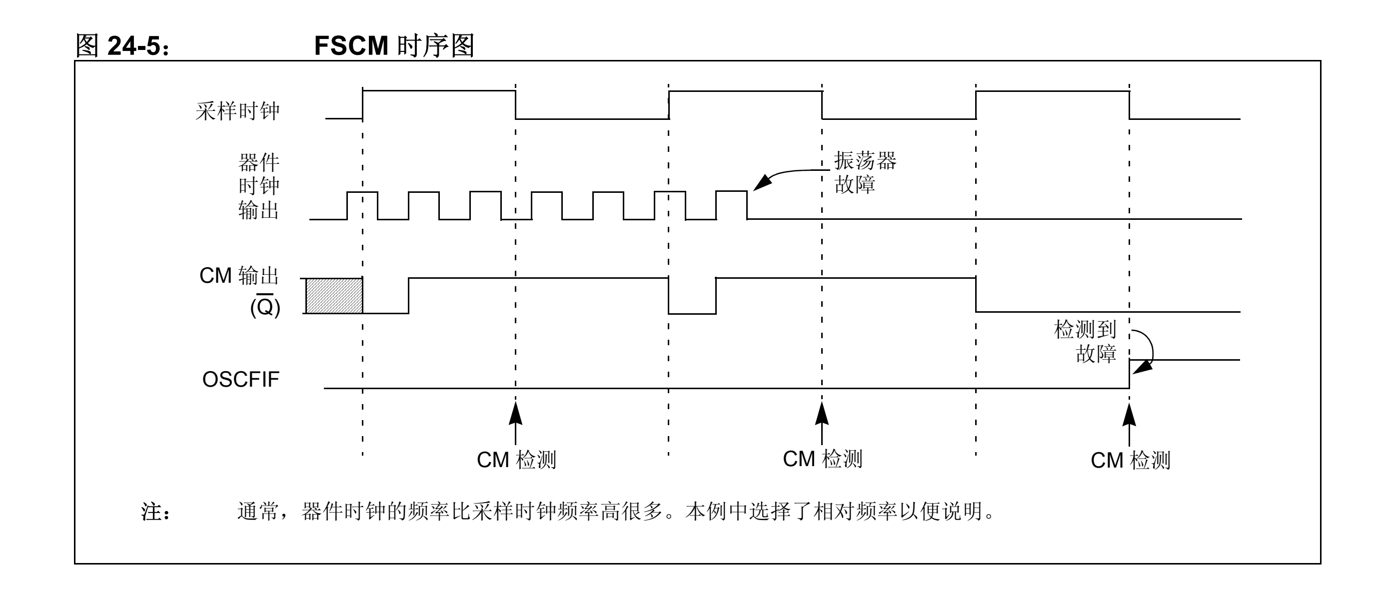
如前所述，当发生时钟故障时，时钟源将切换到 INTRC 时钟源。这可能意味着代码执行速度会发生很大变化。如果使能 WDT 的时候使用的是小预分频值，时钟速度的下降将引起 WDT 超时，随后使器件复位。由于这个原因，故障保护事件也会使 WDT 和后分频器复位，使 WDT 从执行速度发生变化那一刻起开始重新计数，从而避免发生错误超时。

24.5.2 退出故障保护运行模式

器件复位或进入功耗管理模式均可结束故障保护状态。发生复位时，控制器启动在配置寄存器 2H 中指定的主时钟源 （伴有如 OST 或 PLL 定时器等所需的起振延时）。INTRC 振荡器将在主时钟源就绪之前提供系统时钟 （类似于双速启动）。当主时钟源可用时，系统时钟源将切换为主时钟（OSCCON 寄存器中的 OSTS 位将置 1，表明当前使用的是主时钟源）。然后，故障保护时钟监视器恢复对外设时钟的监视。

在启动期间，主时钟源可能永远不能就绪。在这种情况下，器件运行将以INTRC振荡器作为时钟源。OSCCON 寄存器将保持复位状态直到进入功耗管理模式为止。

图 24-5： **FSCM** 时序图



注： 通常，器件时钟的频率比采样时钟频率高很多。本例中选择了相对频率以便说明。

24.5.3 功耗管理模式下的 FSCM 中断

进入功耗管理模式时，时钟多路开关选择由 OSCCON 寄存器选定的时钟源。在该模式下将恢复对功耗管理时钟源的故障保护监视。

如果在功耗管理运行模式下发生了振荡器故障，接下来的操作取决于是否允许了振荡器故障中断。如果允许了（OSCFIF = 1），代码执行将以 INTRC 复用器作为时钟源，并且不会自动转换回发生故障的时钟源。

如果禁止了该中断，处于空闲模式之后产生的中断将使 CPU 开始执行指令，同时以 INTRC 源作为系统时钟源。

24.5.4 POR 或从休眠中唤醒

FSCM 在器件退出上电复位（POR）或低功耗休眠模式后开始检测振荡器故障。当器件主时钟为 EC 或 INTRC 时，监视会在这些事件发生后立即开始。

对于 HS 或 HSPLL 模式，情况会有所不同。由于这类振荡器需要的起振时间可能比FSCM采样时钟的周期长很多，因此可能会检测到假的时钟故障。为了避免这种情况，在此类模式中，内部振荡器电路会被自动配置为器件时钟并一直工作到主时钟稳定下来为止（OST 和 PLL 定时器发生超时）。这与双速启动模式相同。一旦主时钟稳定下来，INTRC 就将重新作为 FSCM 时钟源。

注：	用于防止在POR或从休眠状态唤醒时发生错误中断的逻辑电路，同样也将阻止随后对振荡器故障的检测。通过监视OSTS位，并使用定时程序来确定振荡器起振时间是否过长，可避免这个问题。即使如此，在检测到振荡器故障时也不会将振荡器故障中断标志位置 1。

正如第 **24.4.1** 节 **“使用双速启动时的注意事项”**中所述，在等待系统主时钟稳定的过程中，可以选择另一种时钟配置和另一种功耗管理模式。当选择了新的功耗管理模式时，主时钟将被禁止。

24.6 程序校验和代码保护

对于 PIC18F97J60 系列中的所有器件，片上程序存储空间被视为一个统一的存储区。配置位 CP0 控制该存储区的代码保护。该位阻止外部对程序存储空间的读写。但对正常的代码执行没有直接影响。

24.6.1 配置寄存器保护

有两种方法保护配置寄存器使其免遭破坏性的改写或读取。主要的保护方式是配置位的一次写入功能，该功能阻止对在上电周期内完成编程的位再次进行配置。要阻止不可预见的事件，由于电池故障 （如 ESD 事件）产生的配置位更改将导致奇偶校验错误并触发器件复位。

配置寄存器的数据来自于程序存储器中的闪存配置字。当 CP0 位置 1 时，也将保护器件配置的源数据。

24.7 在线串行编程

PIC18F97J60 系列单片机可以在最终的应用电路中进行串行编程。只需要 5 根线即可实现这一操作，其中时钟线、数据线各一根，其余 3 根分别是电源线、接地线和编程电压线。这允许用户在制造电路板时使用未编程器件，仅在产品交付之前才对单片机进行编程，从而可以使用最新版本的固件或者定制固件。

24.8 在线调试器

将 $\overline{\text{DEBUG}}$ 配置位编程为 `0`，可使能在线调试功能。该功能允许使用 MPLAB® IDE 进行一些简单的调试。当使能了单片机的这项功能时，有些资源就不再是通用的了。表 24-3 给出了后台调试器所需的资源。

表 24-3： 调试器资源

I/O 引脚：	RB6 和 RB7
堆栈：	2 级
程序存储器：	512 字节
数据存储器：	10 字节

25.0 指令集汇总

PIC18F97J60 系列器件具有一个含有 75 条 PIC18 内核指令的标准指令集，和一个含有优化递归或软件堆栈代码的 8 条新指令的扩展指令集。本章后面的部分将讨论扩展指令集。

25.1 标准指令集

标准的 PIC18 指令集与以前的 PIC® 指令集相比，添加了很多增强功能，并保持了易于从其他 PIC 指令集移植的特点。大部分指令为单字指令（16 位），只有 4 条指令是双字指令。

每个单字指令都是一个 16 位字，由操作码 （指明指令类型）和一个或多个操作数 （指定指令操作）组成。

整个指令集具有高度的正交性，可以分为以下 4 种基本类型：

- **字节**操作类指令
- **位**操作类指令
- **立即数**操作类指令
- **控制**操作类指令

表 25-2 为 PIC18 指令集汇总，列出了上述四类指令。表 25-1 给出了操作码字段的说明。

大部分**字节**操作类的指令都含有三种操作数：

1. 文件寄存器 （由 “f”指定）
2. 保存结果的目标寄存器 （由 “d”指定）
3. 被访问存储器 （由 “a”指定）

文件寄存器标识符 “f”指定了指令将会使用哪一个文件寄存器。目标寄存器标识符 “d”指定了操作结果的存放位置。如果 “d”为 0，操作结果存入 WREG 寄存器中。如果 “d”为 1，操作结果存入指令指定的文件寄存器中。

所有**位**操作类指令都含有三种操作数：

1. 文件寄存器 （由 “f”指定）
2. 文件寄存器中的位 （由 “b”指定）
3. 被访问存储器 （由 “a”指定）

位域标识符 “b”选择操作所影响的位的编号，而文件寄存器标识符 “f”则代表这些位所在的寄存器编号。

立即数操作类指令使用以下操作数：

- 要装入文件寄存器中的立即数 （由 “k”指定）
- 要装入立即数的 FSR 寄存器 （由 “f”指定）
- 不需要操作数 （由 “—”指定）

控制类指令可以使用以下操作数：

- 程序存储器地址 （由 “n”指定）
- `CALL` 或 `RETURN` 指令的模式 （由 “s”指定）
- 表读和表写指令的模式 （由 “m”指定）
- 不需要操作数 （由 “—”指定）

除了 4 条双字指令外，所有的指令都是单字指令。双字指令将所需的信息保存在 32 位中。第二个字的高 4 位都是`1`。如果第二个字作为一条指令执行，它会执行`NOP` 指令。

除非条件测试结果为 true 或者指令执行改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，指令执行需要两个指令周期，在第二个指令周期中执行一条 `NOP` 指令。

执行双字指令需要两个指令周期。

每个指令周期由 4 个振荡器周期组成。因此，如果振荡器频率为 4 MHz，正常的指令执行时间为 1 μs。如果条件测试结果为 true 或指令执行改变了程序计数器的值，则该指令的执行时间为 2 μs。双字跳转指令 （如果为 true）的执行则需要 3 μs。

图 25-1 给出了指令的几种通用格式。所有示例均使用 “nnh”来表示十六进制数。

指令集汇总（见表25-2）列出了可被Microchip MPASM™ 汇编器识别的标准指令。

第 25.1.1 节 “标准指令集”中对每个指令进行了介绍。

表 25-1： 操作码字段说明

字段	说明
a	快速操作 RAM 位： a = 0：快速操作 RAM 内的 RAM 单元 （BSR 寄存器被忽略） a = 1：由 BSR 寄存器指定 RAM 存储区
bbb	8 位文件寄存器内的位地址 （0 到 7）。
BSR	存储区选择寄存器。用于选择当前的 RAM 存储区。
C、DC、Z、OV 和 N	ALU 状态位：进位、半进位、全零、溢出和负标志位。
d	目标寄存器选择位： d = 0：结果保存至 WREG 寄存器 d = 1：结果保存至文件寄存器 f
dest	目标寄存器：可以是 WREG 寄存器或指定的文件寄存器地址。
f	8 位文件寄存器地址 （00h 到 FFh），或 2 位 FSR 标识符 （0h 到 3h）。
f_s	12 位文件寄存器地址 （000h 到 FFFh）。这是源地址。
f_d	12 位文件寄存器地址 （000h 到 FFFh）。这是目标地址。
GIE	全局中断允许位。
k	立即数、常数或者标号 （可能是 8 位、 12 位或 20 位的值）。
Label	标号名称。
mm	表读和表写指令的 TBLPTR 寄存器模式。只与表读和表写指令一起使用：
*	不改变寄存器 （如用于表读和表写的 TBLPTR）
*+	后增寄存器 （如用于表读和表写的 TBLPTR）
*-	后减寄存器 （如用于表读和表写的 TBLPTR）
+*	预增寄存器 （如用于表读和表写的 TBLPTR）
n	相对跳转指令的相对地址 （2 的补码），或 Call/Branch 和 Return 指令的直接地址。
PC	程序计数器。
PCL	程序计数器低字节。
PCH	程序计数器高字节。
PCLATH	程序计数器高字节锁存器。
PCLATU	程序计数器最高字节锁存器。
$\overline{PD}$	掉电位。
PRODH	乘积的高字节。
PRODL	乘积的低字节。
s	快速调用 / 返回模式选择位： s = 0：不对影子寄存器进行更新，也不用影子寄存器的内容更新其他寄存器 s = 1：将寄存器的值存入影子寄存器或把影子寄存器中的值载入寄存器 （快速模式）
TBLPTR	21 位表指针 （指向程序存储器地址）。
TABLAT	8 位表锁存器。
$\overline{TO}$	超时溢出位。
TOS	栈顶。
u	未使用或未改变。
WDT	看门狗定时器。
WREG	工作寄存器 （累加器）。
x	无关位 （0 或 1）。汇编器将产生 x = 0 的代码。为了与所有的 Microchip 软件工具兼容，建议使用这种格式。
z_s	对寄存器 （源）进行间接寻址的 7 位偏移量。
z_d	对寄存器 （目标）进行间接寻址的 7 位偏移量。
{ }	可选参数。
[text]	表示变址地址。
(text)	text 的内容。
[expr]<n>	表示由指针 expr 指定的寄存器中的位 n。
→	赋值。
< >	寄存器位域。
∈	表示属于某个集合。
斜体文字	用户定义项 （字体为 Courier）。

 超前信息

图 25-1： 指令的通用格式

面向字节的文件寄存器操作 — 指令示例：`ADDWF MYREG, W, B`

15–10	9	8	7–0
操作码	d	a	f（寄存器地址）

d = 0 表示结果存入 WREG 寄存器
d = 1 表示结果存入文件寄存器（f）
a = 0 强制使用快速操作存储区
a = 1 根据 BSR 选择存储区
f = 8 位文件寄存器地址

字节到字节的传送操作（双字） — 指令示例：`MOVFF MYREG1, MYREG2`

15–12	11–0
操作码	f（源寄存器地址）
1111	f（目标寄存器地址）

f = 12 位文件寄存器地址

面向位的文件寄存器操作 — 指令示例：`BSF MYREG, bit, B`

15–12	11–9	8	7–0
操作码	b（位号）	a	f（寄存器地址）

b = 占 3 位，表示文件寄存器（f）中位的位置
a = 0 强制使用快速操作存储区
a = 1 根据 BSR 选择存储区
f = 8 位文件寄存器地址

立即数操作 — 指令示例：`MOVLW 7Fh`

15–8	7–0
操作码	k（立即数）

k = 8 位立即值

控制操作

CALL、GOTO 和跳转类操作指令

指令示例：`GOTO Label`

15–8	7–0
操作码	n<7:0>（立即数）

15–12	11–0
1111	n<19:8>（立即数）

k = 20 位立即数的值

指令示例：`CALL MYFUNC`

15–9	8	7–0
操作码	S	n<7:0>（立即数）

15–12	11–0
1111	n<19:8>（立即数）

S = 快速位

指令示例：`BRA MYFUNC`

15–11	10–0
操作码	n<10:0>（立即数）

指令示例：`BC MYFUNC`

15–8	7–0
操作码	n<7:0>（立即数）

表 25-2： PIC18F97J60 系列指令集

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位	注
面向字节的操作类指令									
ADDWF	f, d, a	WREG 与 f 相加	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	WREG 与 f 带进位相加	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	WREG 和 f 作与运算	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	将 f 清零	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	对 f 取反	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	将 f 与 WREG 作比较，相等则跳过	1 (2 或 3)	0110	001a	ffff	ffff	无	4
CPFSGT	f, a	将 f 与 WREG 作比较，大于则跳过	1 (2 或 3)	0110	010a	ffff	ffff	无	4
CPFSLT	f, a	将 f 与 WREG 作比较，小于则跳过	1 (2 或 3)	0110	000a	ffff	ffff	无	1, 2
DECF	f, d, a	f 减 1	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	f 减 1，为 0 则跳过	1 (2 或 3)	0010	11da	ffff	ffff	无	1, 2, 3, 4
DCFSNZ	f, d, a	f 减 1，非 0 则跳过	1 (2 或 3)	0100	11da	ffff	ffff	无	1, 2
INCF	f, d, a	f 加 1	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	f 加 1，为 0 则跳过	1 (2 或 3)	0011	11da	ffff	ffff	无	4
INFSNZ	f, d, a	f 加 1，非 0 则跳过	1 (2 或 3)	0100	10da	ffff	ffff	无	1, 2
IORWF	f, d, a	WREG 和 f 作或运算	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	移动 f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	f_s（源）地址装入第一个字 f_d（目标）地址装入第二个字	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	无	
MOVWF	f, a	将 WREG 移入 f	1	0110	111a	ffff	ffff	无	
MULWF	f, a	WREG 乘以 f	1	0000	001a	ffff	ffff	无	1, 2
NEGF	f, a	对 f 取补	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	f 带进位循环左移	1	0011	01da	ffff	ffff	C, Z, N	1, 2
RLNCF	f, d, a	f 循环左移 （不带进位）	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	f 带进位循环右移	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	f 循环右移 （不带进位）	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	将 f 的内容置为全 1	1	0110	100a	ffff	ffff	无	1, 2
SUBFWB	f, d, a	WREG 减去 f （带借位）	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	f 减去 WREG	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	f 减去 WREG （带借位）	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	将 f 中的两个半字节进行交换	1	0011	10da	ffff	ffff	无	4
TSTFSZ	f, a	测试 f，为 0 则跳过	1 (2 或 3)	0110	011a	ffff	ffff	无	1, 2
XORWF	f, d, a	WREG 和 f 作异或运算	1	0001	10da	ffff	ffff	Z, N	

注 1： PORT 寄存器的值随端口状态的变化而不断修改（例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 `1`，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 `0`。

2： 当对 TMR0 寄存器执行该指令（并且 d = `1`）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器（PC）被修改或者条件检测为 true，则该指令需要两个周期。第二个周期执行一条 `NOP` 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 `NOP` 指令执行。这将确保所有程序存储器单元内存储的都是合法的指令。

表 25-2： PIC18F97J60 系列指令集 （续）

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位	注
面向位的操作类指令									
BCF	f, b, a	将 f 寄存器中的某位清零	1	1001	bbba	ffff	ffff	无	1, 2
BSF	f, b, a	将 f 寄存器中的某位置 1	1	1000	bbba	ffff	ffff	无	1, 2
BTFSC	f, b, a	测试 f 中的某位，为 0 则跳过	1 (2 或 3)	1011	bbba	ffff	ffff	无	3, 4
BTFSS	f, b, a	测试 f 中的某位，为 1 则跳过	1 (2 或 3)	1010	bbba	ffff	ffff	无	3, 4
BTG	f, b, a	将 f 中的某位取反	1	0111	bbba	ffff	ffff	无	1, 2
控制类指令									
BC	n	进位则跳转	1 (2)	1110	0010	nnnn	nnnn	无	
BN	n	为负则跳转	1 (2)	1110	0110	nnnn	nnnn	无	
BNC	n	无进位则跳转	1 (2)	1110	0011	nnnn	nnnn	无	
BNN	n	不为负则跳转	1 (2)	1110	0111	nnnn	nnnn	无	
BNOV	n	不溢出则跳转	1 (2)	1110	0101	nnnn	nnnn	无	
BNZ	n	不为零则跳转	1 (2)	1110	0001	nnnn	nnnn	无	
BOV	n	溢出则跳转	1 (2)	1110	0100	nnnn	nnnn	无	
BRA	n	无条件跳转	2	1101	0nnn	nnnn	nnnn	无	
BZ	n	为零则跳转	1 (2)	1110	0000	nnnn	nnnn	无	
CALL	n, s	调用子程序 第一个字	2	1110	110s	kkkk	kkkk	无	
		第二个字		1111	kkkk	kkkk	kkkk		
CLRWDT	—	将看门狗定时器清零	1	0000	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	对 WREG 进行十进制调整	1	0000	0000	0000	0111	C	
GOTO	n	跳转到地址 第一个字	2	1110	1111	kkkk	kkkk	无	
		第二个字		1111	kkkk	kkkk	kkkk		
NOP	—	空操作	1	0000	0000	0000	0000	无	
NOP	—	空操作	1	1111	xxxx	xxxx	xxxx	无	4
POP	—	返回堆栈栈顶 （TOS）出栈	1	0000	0000	0000	0110	无	
PUSH	—	返回堆栈栈顶 （TOS）进栈	1	0000	0000	0000	0101	无	
RCALL	n	相对调用	2	1101	1nnn	nnnn	nnnn	无	
RESET		用软件使器件复位	1	0000	0000	1111	1111	全部	
RETFIE	s	中断返回允许	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
RETURN	s	从子程序返回	2	0000	0000	0001	001s	无	
SLEEP	—	进入待机模式	1	0000	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	

注 **1：** PORT 寄存器的值随端口状态的变化而不断修改 （例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 `1`，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 `0`。

2： 当对 TMR0 寄存器执行该指令 （并且 d = `1`）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，则该指令需要两个周期。第二个周期执行一条 `NOP` 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 `NOP` 指令执行。这将确保所有程序存储器单元内存储的都是合法的指令。

表 25-2： PIC18F97J60 系列指令集 （续）

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位	注
立即数操作类指令									
ADDLW	k	WREG 与立即数相加	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	WREG 和立即数进行与运算	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	WREG 和立即数进行或运算	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	移动立即数 （12 位）第二个字到 FSR （f）第一个字	2	1110 1111	1110 0000	00ff kkkk	kkkk kkkk	无	
MOVLB	k	将立即数移入 BSR<3:0>	1	0000	0001	0000	kkkk	无	
MOVLW	k	将立即数移入 WREG	1	0000	1110	kkkk	kkkk	无	
MULLW	k	WREG 和立即数相乘	1	0000	1101	kkkk	kkkk	无	
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
SUBLW	k	立即数减去 WREG	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	WREG 和立即数进行异或运算	1	0000	1010	kkkk	kkkk	Z, N	
数据存储器 ↔ 程序存储器操作									
TBLRD*		表读	2	0000	0000	0000	1000	无	
TBLRD*+		后增表读		0000	0000	0000	1001	无	
TBLRD*-		后减表读		0000	0000	0000	1010	无	
TBLRD+*		预增表读		0000	0000	0000	1011	无	
TBLWT*		表写	2	0000	0000	0000	1100	无	
TBLWT*+		后增表写		0000	0000	0000	1101	无	
TBLWT*-		后减表写		0000	0000	0000	1110	无	
TBLWT+*		预增表写		0000	0000	0000	1111	无	

注 1： PORT 寄存器的值随端口状态的变化而不断修改（例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 1，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 0。

2： 当对 TMR0 寄存器执行该指令（并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储器单元内存储的都是合法的指令。

25.1.1 标准指令集

ADDLW **W 与立即数相加**

语法：	ADDLW k
操作数：	0 ≤ k ≤ 255
操作：	(W) + k → W
受影响的状态位：	N、 OV、 C、 DC 和 Z
机器码：	0000 1111 kkkk kkkk
说明：	将 W 寄存器的内容与 8 位立即数 k 相加，结果存储在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例： ADDLW 15h

执行指令前
W = 10h
执行指令后
W = 25h

ADDWF **W 与 f 寄存器相加**

语法：	ADDWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(W) + (f) → dest
受影响的状态位：	N、 OV、 C、 DC 和 Z
机器码：	0010 01da ffff ffff
说明：	将 W 的内容与 f 寄存器的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： ADDWF REG, 0, 0

执行指令前
W = 17h
REG = 0C2h
执行指令后
W = 0D9h
REG = 0C2h

注：	所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令语法将变为：{label} 指令参数。

ADDWFC **W 与 f 带进位相加**

语法： ADDWFC f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： $(W) + (f) + (C) \rightarrow dest$

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0010	00da	ffff	ffff

说明： 将 W 的内容、进位标志位与寄存器 f 的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存储在寄存器 f 中。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例： `ADDWFC REG, 0, 1`

执行指令前
进位标志位 = 1
REG = 02h
W = 4Dh
执行指令后
进位标志位 = 0
REG = 02h
W = 50h

ANDLW **立即数和 W 寄存器作逻辑与运算**

语法： ANDLW k

操作数： $0 \le k \le 255$

操作： (W) .AND. $k \rightarrow W$

受影响的状态位： N 和 Z

机器码：

0000	1011	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 进行逻辑与运算。结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例： `ANDLW 05Fh`

执行指令前
W = A3h
执行指令后
W = 03h

ANDWF　　将 W 和 f 作逻辑与运算

语法：　ANDWF　f {,d {,a}}

操作数：　$0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作：　(W) .AND.(f) $\rightarrow$ dest

受影响的状态位：　N 和 Z

机器码：

0001	01da	ffff	ffff

说明：　将 W 的内容与寄存器 f 的内容进行逻辑与运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例：　`ANDWF　REG, 0, 0`

执行指令前
　W　=　17h
　REG　=　C2h
执行指令后
　W　=　02h
　REG　=　C2h

BC　　进位则跳转

语法：　BC　n

操作数：　$-128 \le n \le 127$

操作：　如果进位标志位为 1
(PC) + 2 + 2n $\rightarrow$ PC

受影响的状态位：　无

机器码：

1110	0010	nnnn	nnnn

说明：　如果进位标志位为 1，程序将跳转。

"2n"（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数：　1

指令周期数：　1(2)

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例：　`HERE　BC　5`

执行指令前
　PC　=　地址（HERE）
执行指令后
　如果进位标志位 =　1；
　　PC　=　地址（HERE + 12）
　如果进位标志位 =　0；
　　PC　=　地址（HERE + 2）

BCF 将 f 寄存器中的某位清零

语法： BCF f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 0 → f<b>

受影响的状态位： 无

机器码：

1001	bbba	ffff	ffff

说明： 将寄存器 f 中的位 b 清零。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： BCF FLAG_REG, 7, 0

执行指令前
FLAG_REG = C7h
执行指令后
FLAG_REG = 47h

BN 为负则跳转

语法： BN n

操作数： -128 ≤ n ≤ 127

操作： 如果负标志位为 1
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0110	nnnn	nnnn

说明： 如果负标志位为 1，程序将跳转。

"2n"（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 1(2)

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BN Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果负标志位 = 1；
PC = 地址（Jump）
如果负标志位 = 0；
PC = 地址（HERE + 2）

BNC 无进位则跳转

语法：	BNC n
操作数：	$-128 \le n \le 127$
操作：	如果进位标志位为 0 $(PC) + 2 + 2n \rightarrow PC$
受影响的状态位：	无
机器码：	1110 | 0011 | nnnn | nnnn
说明：	如果进位标志位为 0，程序将跳转。 “2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。
指令字数：	1
指令周期数：	1(2)

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNC Jump

执行指令前
- PC = 地址（HERE）

执行指令后
- 如果进位标志位 = 0；
  - PC = 地址（Jump）
- 如果进位标志位 = 1；
  - PC = 地址（HERE + 2）

BNN 不为负则跳转

语法：	BNN n
操作数：	$-128 \le n \le 127$
操作：	如果负标志位为 0 $(PC) + 2 + 2n \rightarrow PC$
受影响的状态位：	无
机器码：	1110 | 0111 | nnnn | nnnn
说明：	如果负标志位为 0，程序将跳转。 “2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。
指令字数：	1
指令周期数：	1(2)

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNN Jump

执行指令前
- PC = 地址（HERE）

执行指令后
- 如果负标志位 = 0；
  - PC = 地址（Jump）
- 如果负标志位 = 1；
  - PC = 地址（HERE + 2）

BNOV 不溢出则跳转

语法： BNOV n

操作数： $-128 \leq n \leq 127$

操作： 如果溢出标志位为 0
$(PC) + 2 + 2n \rightarrow PC$

受影响的状态位： 无

机器码：

1110	0101	nnnn	nnnn

说明： 如果溢出标志位为 0，程序将跳转。
“2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 1(2)

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNOV Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果溢出标志位 = 0；
PC = 地址（Jump）
如果溢出标志位 = 1；
PC = 地址（HERE + 2）

BNZ 不为零则跳转

语法： BNZ n

操作数： $-128 \leq n \leq 127$

操作： 如果全零标志位为 0
$(PC) + 2 + 2n \rightarrow PC$

受影响的状态位： 无

机器码：

1110	0001	nnnn	nnnn

说明： 如果全零标志位为 0，程序将跳转。
“2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 1(2)

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNZ Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果全零标志位 = 0；
PC = 地址（Jump）
如果全零标志位 = 1；
PC = 地址（HERE + 2）

 超前信息

BRA 无条件跳转

语法： BRA n

操作数： -1024 ≤ n ≤ 1023

操作： (PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1101	0nnn	nnnn	nnnn

说明： “2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例： HERE BRA Jump

执行指令前
PC = 地址 （HERE）

执行指令后
PC = 地址 （Jump）

BSF 将 f 寄存器中的某位置 1

语法： BSF f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 1 → f<b>

受影响的状态位： 无

机器码：

1000	bbba	ffff	ffff

说明： 将寄存器 f 的位 b 置 1。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： BSF FLAG_REG, 7, 1

执行指令前
FLAG_REG = 0Ah

执行指令后
FLAG_REG = 8Ah

BTFSC 测试寄存器中的位，为 0 则跳过

语法：	BTFSC f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]
操作：	如果 (f<b>) = 0，则跳过
受影响的状态位：	无
机器码：	1011 bbba ffff ffff
说明：	如果寄存器 f 的位 b 为 0，则跳过下一条指令。即在 b 位为 0 时，丢弃下一条指令（执行当前指令期间取的指令）转而执行一条 NOP 指令，使该指令变成双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1(2) 注： 如果跳过的指令后面跟有 2 字指令，则执行 BTFSC 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址 （HERE）
执行指令后
如果 FLAG<1> = 0；
PC = 地址 （TRUE）
如果 FLAG<1> = 1；
PC = 地址 （FALSE）

BTFSS 测试寄存器中的位，为 1 则跳过

语法：	BTFSS f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b < 7 a ∈ [0,1]
操作：	如果 (f<b>) = 1，则跳过
受影响的状态位：	无
机器码：	1010 bbba ffff ffff
说明：	如果寄存器 f 的位 b 为 1，则跳过下一条指令。即在 b 位为 1 时，丢弃下一条指令（执行当前指令期间取的指令）转而执行一条 NOP 指令，使该指令变成双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1(2) 注： 如果跳过的指令后面跟有 2 字指令，则执行 BTFSS 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址 （HERE）
执行指令后
如果 FLAG<1> = 0；
PC = 地址 （FALSE）
如果 FLAG<1> = 1；
PC = 地址 （TRUE）

BTG **将 f 中的某位取反**

语法：	BTG f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b < 7 a ∈ [0,1]
操作：	$(\overline{f<b>}) \rightarrow f<b>$
受影响的状态位：	无
机器码：	`0111` `bbba` `ffff` `ffff`
说明：	将数据存储单元 f 中的位 b 取反。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `BTG PORTC, 4, 0`

执行指令前：
PORTC = `0111 0101` [75h]
执行指令后：
PORTC = `0110 0101` [65h]

BOV **溢出则跳转**

语法：	BOV n
操作数：	-128 ≤ n ≤ 127
操作：	如果溢出标志位为 1 $(PC) + 2 + 2n \rightarrow PC$
受影响的状态位：	无
机器码：	`1110` `0100` `nnnn` `nnnn`
说明：	如果溢出标志位为 1，程序将跳转。 "2n"（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。
指令字数：	1
指令周期数：	1(2)

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： `HERE BOV Jump`

执行指令前
PC = 地址 （`HERE`）
执行指令后
如果溢出标志位 = 1；
PC = 地址 （`Jump`）
如果溢出标志位 = 0；
PC = 地址 （`HERE + 2`）

BZ　为零则跳转

语法：BZ n

操作数：-128 ≤ n ≤ 127

操作：如果全零标志位为 1
(PC) + 2 + 2n → PC

受影响的状态位：无

机器码：

1110	0000	nnnn	nnnn

说明：如果全零标志位为 1，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数：1

指令周期数：1(2)

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例：　HERE　BZ　Jump

执行指令前
PC = 地址（HERE）

执行指令后
如果全零标志位 = 1；
PC = 地址（Jump）
如果全零标志位 = 0；
PC = 地址（HERE + 2）

CALL　调用子程序

语法：CALL k {,s}

操作数：0 ≤ k ≤ 1048575
s ∈ [0,1]

操作：(PC) + 4 → TOS，
k → PC<20:1>，
如果 s = 1
(W) → WS，
(STATUS) → STATUSS，
(BSR) → BSRS

受影响的状态位：无

机器码：

第一个字（k<7:0>）	1110	110s	$k_7$kkk	kkkk$_0$
第二个字（k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明：可在整个 2MB 的存储器范围内进行子程序调用。首先，将返回地址（PC + 4）压入返回堆栈。如果 s = 1，还会将 W、STATUS 和 BSR 寄存器的内容存入它们各自的影子寄存器 WS、STATUSS 和 BSRS。如果 s = 0，将不会进行任何更新（默认）。然后，将 20 位的值 k 装入 PC<20:1>。CALL 是一条双周期指令。

指令字数：2

指令周期数：2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>	将 PC 压入堆栈	读立即数 k<19:8>，写入 PC
空操作	空操作	空操作	空操作

示例：　HERE　CALL　THERE,1

执行指令前
PC = 地址（HERE）

执行指令后
PC = 地址（THERE）
TOS = 地址（HERE + 4）
WS = W
BSRS = BSR
STATUSS = STATUS

CLRF **将 f 清零**

语法：	CLRF f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	000h → f 1 → Z
受影响的状态位：	Z
机器码：	0110 101a ffff ffff
说明：	清零指定寄存器的内容。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： CLRF FLAG_REG,1

执行指令前
FLAG_REG = 5Ah
执行指令后
FLAG_REG = 00h

CLRWDT **将看门狗定时器清零**

语法：	CLRWDT
操作数：	无
操作：	000h → WDT， 000h → WDT 后分频器， 1 → $\overline{TO}$， 1 → $\overline{PD}$
受影响的状态位：	$\overline{TO}$ 和 $\overline{PD}$
机器码：	0000 0000 0000 0100
说明：	CLRWDT指令复位看门狗定时器及其后分频器。状态位 $\overline{TO}$ 和 $\overline{PD}$ 置 1。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

示例： CLRWDT

执行指令前
WDT 计数器 = ?
执行指令后
WDT 计数器 = 00h
WDT 后分频器 = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF 对 f 取反

语法：	COMF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	$\overline{(f)} \rightarrow$ dest
受影响的状态位：	N 和 Z
机器码：	0001 | 11da | ffff | ffff
说明：	将寄存器 f 的内容取反。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： COMF REG, 0, 0

执行指令前
REG = 13h
执行指令后
REG = 13h
W = ECh

CPFSEQ 比较 f 和 W，如果 f = W 则跳过

语法：	CPFSEQ f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	(f) – (W), 如果 (f) = (W)，则跳过 （无符号比较）
受影响的状态位：	无
机器码：	0110 | 001a | ffff | ffff
说明：	通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容作比较。 如果 f = W，则所取的指令被丢弃并执行一条 NOP 指令，使这个指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1(2) 注： 如果跳过的指令后面跟有 2 字指令，则执行 CPFSEQ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     CPFSEQ REG, 0
NEQUAL   :
EQUAL    :
```

执行指令前
PC 地址 = HERE
W = ?
REG = ?
执行指令后
如果 REG = W；
PC = 地址（EQUAL）
如果 REG ≠ W；
PC = 地址（NEQUAL）

CPFSGT **比较 f 和 W，如果 f > W 则跳过**

语法：	CPFSGT f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	(f) – (W), 如果 (f) > (W)，则跳过 （无符号比较）
受影响的状态位：	无
机器码：	`0110` `010a` `ffff` `ffff`
说明：	通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容作比较。 如果 f > W，则所取的指令被丢弃并执行一条 NOP 指令，使这个指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1(2) **注：** 如果跳过的指令后面跟有 2 字指令，则执行 CPFSGT 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```

执行指令前
PC = 地址 （HERE）
W = ?

执行指令后
如果 REG > W；
PC = 地址 （GREATER）
如果 REG ≤ W；
PC = 地址 （NGREATER）

CPFSLT **比较 f 和 W，如果 f < W 则跳过**

语法：	CPFSLT f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	(f) – (W), 如果 (f) < (W)，则跳过 （无符号比较）
受影响的状态位：	无
机器码：	`0110` `000a` `ffff` `ffff`
说明：	通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容作比较。 如果 f < W，则所取的指令被丢弃并执行一条 NOP 指令，使这个指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
指令字数：	1
指令周期数：	1(2) **注：** 如果跳过的指令后面跟有 2 字指令，则执行 CPFSLT 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

执行指令前
PC = 地址 （HERE）
W = ?

执行指令后
如果 REG < W；
PC = 地址 （LESS）
如果 REG ≥ W；
PC = 地址 （NLESS）

DAW 对 W 寄存器进行十进制调整

语法：	DAW
操作数：	无
操作：	如果 [W<3:0> > 9] 或 [DC = 1]，则 (W<3:0>) + 6 → W<3:0>； 否则 (W<3:0>) → W<3:0> 如果 [W<7:4> > 9] 或 [C = 1]，则 (W<7:4>) + 6 → W<7:4>； C = 1； 否则 (W<7:4>) → W<7:4>
受影响的状态位：	C
机器码：	0000 0000 0000 0111
说明：	DAW指令调整W寄存器内的8位值，即前两个压缩 BCD 格式的变量之和，并产生一个正确的压缩 BCD 格式结果。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 W	处理数据	写 W

例 1： DAW

执行指令前
W = A5h
C = 0
DC = 0
执行指令后
W = 05h
C = 1
DC = 0

例 2：

执行指令前
W = CEh
C = 0
DC = 0
执行指令后
W = 34h
C = 1
DC = 0

DECF f 减 1

语法：	DECF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) – 1 → dest
受影响的状态位：	C、 DC、 N、 OV 和 Z
机器码：	0000 01da ffff ffff
说明：	将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例： DECF CNT, 1, 0

执行指令前
CNT = 01h
Z = 0
执行指令后
CNT = 00h
Z = 1

DECFSZ **f 减 1，为 0 则跳过**

语法：	DECFSZ f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) – 1 → dest， 如果结果 = 0 则跳过
受影响的状态位：	无

机器码：

0010	11da	ffff	ffff

说明：

将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果结果为 0，则丢弃已取的指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1(2)

注： 如果跳过的指令后面跟有 2 字指令，则执行 DECFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE      DECFSZ   CNT, 1, 1
          GOTO     LOOP
CONTINUE
```

执行指令前
PC = 地址 （HERE）
执行指令后
CNT = CNT – 1
如果 CNT = 0；
PC = 地址 （CONTINUE）
如果 CNT ≠ 0；
PC = 地址 （HERE + 2）

DCFSNZ **f 减 1，非 0 则跳过**

语法：	DCFSNZ f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) – 1 → dest， 如果结果 ≠ 0 则跳过
受影响的状态位：	无

机器码：

0100	11da	ffff	ffff

说明：

将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果结果不为 0，则丢弃已取的指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1(2)

注： 如果跳过的指令后面跟有 2 字指令，则执行 DCFSNZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    DCFSNZ  TEMP, 1, 0
ZERO    :
NZERO   :
```

执行指令前
TEMP = ?
执行指令后
TEMP = TEMP – 1，
如果 TEMP = 0；
PC = 地址 （ZERO）
如果 TEMP ≠ 0；
PC = 地址 （NZERO）

GOTO **无条件跳转**

语法： GOTO k

操作数： 0 ≤ k ≤ 1048575

操作： k → PC<20:1>

受影响的状态位： 无

机器码：

第一个字 （k<7:0>）	1110	1111	$k_7$kkk	kkkk$_0$
第二个字 （k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： GOTO 指令允许无条件跳转到整个 2 MB 存储器范围中的任何位置。将 20 位值 k 装入 PC<20:1>。 GOTO 始终为双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>	空操作	读立即数 k<19:8>，写入 PC
空操作	空操作	空操作	空操作

示例： GOTO THERE

执行指令后
PC = 地址 （THERE）

INCF **f 加 1**

语法： INCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) + 1 → dest

受影响的状态位： C、 DC、 N、 OV 和 Z

机器码：

0010	10da	ffff	ffff

说明： 将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： INCF CNT, 1, 0

执行指令前
CNT = FFh
Z = 0
C = ?
DC = ?

执行指令后
CNT = 00h
Z = 1
C = 1
DC = 1

INCFSZ　　f 加 1，为 0 则跳过

语法：	INCFSZ　f {,d {,a}}
操作数：	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
操作：	$(f) + 1 \rightarrow$ dest， 如果结果 = 0 则跳过
受影响的状态位：	无
机器码：	0011 　11da 　ffff 　ffff
说明：	将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回 f 寄存器。（默认） 如果结果为 0，则丢弃已取的指令转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1(2) **注：** 如果跳过的指令后面跟有 2 字指令，则执行 INCFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     INCFSZ   CNT, 1, 0
NZERO    :
ZERO     :
```

执行指令前
PC = 地址 （HERE）
执行指令后
CNT = CNT + 1
如果 CNT = 0；
PC = 地址 （ZERO）
如果 CNT ≠ 0；
PC = 地址 （NZERO）

INFSNZ　　f 加 1，非 0 则跳过

语法：	INFSNZ　f {,d {,a}}
操作数：	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
操作：	$(f) + 1 \rightarrow$ dest， 如果结果 ≠ 0 则跳过
受影响的状态位：	无
机器码：	0100 　10da 　ffff 　ffff
说明：	将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果结果不为 0，则丢弃已取的指令转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1(2) **注：** 如果跳过的指令后面跟有 2 字指令，则执行 INFSNZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     INFSNZ   REG, 1, 0
ZERO
NZERO
```

执行指令前
PC = 地址 （HERE）
执行指令后
REG = REG + 1
如果 REG ≠ 0；
PC = 地址 （NZERO）
如果 REG = 0；
PC = 地址 （ZERO）

IORLW **将立即数与 W 作逻辑或运算**

语法： IORLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .OR. k → W

受影响的状态位： N 和 Z

机器码：

0000	1001	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 进行逻辑或运算。结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例： `IORLW 35h`

执行指令前
W = 9Ah

执行指令后
W = BFh

IORWF **将 W 与 f 作逻辑或运算**

语法： IORWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .OR.(f) → dest

受影响的状态位： N 和 Z

机器码：

0001	00da	ffff	ffff

说明： 将 W 的内容与寄存器 f 的内容进行逻辑或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： `IORWF RESULT, 0, 1`

执行指令前
RESULT = 13h
W = 91h

执行指令后
RESULT = 13h
W = 93h

 超前信息

LFSR 载入 FSR

语法：	LFSR f, k
操作数：	0 ≤ f ≤ 2 0 ≤ k ≤ 4095
操作：	k → FSRf
受影响的状态位：	无
机器码：	见下表
说明：	将 12 位立即数 k 载入 f 所指向的文件选择寄存器。
指令字数：	2
指令周期数：	2

机器码：

1110	1110	00ff	k_{11}kkk
1111	0000	$k_7$kkk	kkkk

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k 的 MSB	处理数据	将立即数 k 的 MSB 写入 FSRfH
译码	读立即数 k 的 LSB	处理数据	将立即数 k 的 LSB 写入 FSRfL

示例： `LFSR 2, 3ABh`

执行指令后
FSR2H = 03h
FSR2L = ABh

MOVF 移动 f

语法：	MOVF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	f → dest
受影响的状态位：	N 和 Z
机器码：	0101 | 00da | ffff | ffff
说明：	根据 d 的状态，将寄存器 f 的内容移入目标单元。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。f 可以为 256 字节存储区中的任何单元。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写 W

示例： `MOVF REG, 0, 0`

执行指令前
REG = 22h
W = FFh
执行指令后
REG = 22h
W = 22h

MOVFF　将源寄存器的内容移到目标寄存器

语法：　MOVFF f_s,f_d

操作数：　$0 \le f_s \le 4095$
$0 \le f_d \le 4095$

操作：　$(f_s) \rightarrow f_d$

受影响的状态位：　无

机器码：

第一个字（源）	1100	ffff	ffff	$ffff_s$
第二个字（目标）	1111	ffff	ffff	$ffff_d$

说明：　将源寄存器 f_s 的内容移入目标寄存器 f_d。源寄存器 f_s 可以是 4096 字节数据空间（000h 到 FFFh）中的任何单元，目标寄存器 f_d 也可以是 000h 到 FFFh 中的任何单元。

源或目标寄存器都可以是 W（这是个有用的特例）。

MOVFF 指令对于将数据存储单元中的内容移入外设寄存器（如发送缓冲器或 I/O 端口）的场合非常有用。

MOVFF 指令不能使用 PCL、TOSU、TOSH 或 TOSL 作为目标寄存器。

指令字数：　2

指令周期数：　2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f（源寄存器）	处理数据	空操作
译码	空操作 无效读取	空操作	写寄存器 f（目标寄存器）

示例：　MOVFF　REG1, REG2

执行指令前
REG1 = 33h
REG2 = 11h
执行指令后
REG1 = 33h
REG2 = 33h

MOVLB　将立即数移入 BSR 的低半字节

语法：　MOVLW k

操作数：　$0 \le k \le 255$

操作：　$k \rightarrow$ BSR

受影响的状态位：　无

机器码：

0000	0001	kkkk	kkkk

说明：　将 8 位立即数 k 装入存储区选择寄存器（BSR）。不管 k_7:k_4 的值如何，BSR<7:4> 的值将始终保持为 0。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	将立即数 k 写入 BSR

示例：　MOVLB　5

执行指令前
BSR 寄存器 = 02h
执行指令后
BSR 寄存器 = 05h

MOVLW 将立即数移入 W

语法：	MOVLW k
操作数：	$0 \le k \le 255$
操作：	$k \rightarrow W$
受影响的状态位：	无
机器码：	0000 1110 kkkk kkkk
说明：	将 8 位立即数 k 装入 W。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例： `MOVLW 5Ah`

执行指令后
W = 5Ah

MOVWF 将 W 的内容移入 f

语法：	MOVWF f {,a}
操作数：	$0 \le f \le 255$ $a \in [0,1]$
操作：	$(W) \rightarrow f$
受影响的状态位：	无
机器码：	0110 111a ffff ffff
说明：	将 W 寄存器中的数据移入寄存器 f。f 可以是 256 字节存储区中的任何地址单元。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `MOVWF REG, 0`

执行指令前
W = 4Fh
REG = FFh
执行指令后
W = 4Fh
REG = 4Fh

MULLW 将立即数与 W 中的内容相乘

语法:	MULLW k
操作数:	0 ≤ k ≤ 255
操作:	(W) x k → PRODH:PRODL
受影响的状态位:	无
机器码:	0000 1101 kkkk kkkk
说明:	将 W 的内容与 8 位立即数 k 进行无符号的乘法运算。 16 位的结果存储在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。 W 的内容不改变。 所有状态标志位都不受影响。 请注意此操作不可能发生溢出或进位。结果有可能为零，但不会被检测到。
指令字数:	1
指令周期数:	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写寄存器 PRODH: PRODL

示例： `MULLW 0C4h`

执行指令前
W = E2h
PRODH = ?
PRODL = ?

执行指令后
W = E2h
PRODH = ADh
PRODL = 08h

MULWF 将 W 与 f 的内容相乘

语法:	MULWF f {,a}
操作数:	0 ≤ f ≤ 255 a ∈ [0,1]
操作:	(W) x (f) → PRODH:PRODL
受影响的状态位:	无
机器码:	0000 001a ffff ffff
说明:	将 W 的内容与寄存器单元 f 的内容执行无符号的乘法运算。运算的 16 位结果保存在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。 W 和 f 的内容都不改变。 所有状态标志位都不受影响。 请注意此操作不可能发生溢出或进位。结果有可能为零，但不会被检测到。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数:	1
指令周期数:	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 PRODH: PRODL

示例： `MULWF REG, 1`

执行指令前
W = C4h
REG = B5h
PRODH = ?
PRODL = ?

执行指令后
W = C4h
REG = B5h
PRODH = 8Ah
PRODL = 94h

NEGF 对 f 取补

语法： NEGF f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

操作： $(\bar{f}) + 1 \rightarrow f$

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0110	110a	ffff	ffff

说明： 用二进制补码对单元 f 取补，结果存储在数据存储单元 f 中。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `NEGF REG, 1`

执行指令前
REG = 0011 1010 [3Ah]
执行指令后
REG = 1100 0110 [C6h]

NOP 空操作

语法： NOP

操作数： 无

操作： 空操作

受影响的状态位： 无

机器码：

0000	0000	0000	0000
1111	xxxx	xxxx	xxxx

说明： 不执行任何操作。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

示例：

无。

POP 弹出返回堆栈栈顶的内容

语法： POP

操作数： 无

操作： (TOS) → 丢弃

受影响的状态位： 无

机器码：

0000	0000	0000	0110

说明： 从返回堆栈弹出 TOS 值并丢弃。然后，前一个压入返回堆栈的值成为 TOS 值。此指令可以让用户正确管理返回堆栈，从而实现软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	弹出 TOS 值	空操作

示例：

```
POP
GOTO      NEW
```

执行指令前

TOS = 0031A2h

堆栈 （下一级） = 014332h

执行指令后

TOS = 014332h

PC = NEW

PUSH 将数据压入返回堆栈栈顶

语法： PUSH

操作数： 无

操作： (PC + 2) → TOS

受影响的状态位： 无

机器码：

0000	0000	0000	0101

说明： PC + 2 的值被压入返回堆栈的栈顶。原先的 TOS 值被压入堆栈的下一级。此指令允许通过修改 TOS 并将其压入返回堆栈来实现软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	将 PC + 2 压入返回堆栈	空操作	空操作

示例：

```
PUSH
```

执行指令前

TOS = 345Ah

PC = 0124h

执行指令后

PC = 0126h

TOS = 0126h

堆栈 （下一级） = 345Ah

超前信息

RCALL **相对调用**

语法：	RCALL n
操作数：	$-1024 \leq n \leq 1023$
操作：	(PC) + 2 → TOS， (PC) + 2 + 2n → PC
受影响的状态位：	无
机器码：	1101 1nnn nnnn nnnn
说明：	从当前地址跳转 （最多 1K）来调用子程序。首先，将返回地址 （PC + 2）压入返回堆栈。然后，将 “2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 n 将 PC 压入堆栈	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例： HERE RCALL Jump

执行指令前
PC = 地址 （HERE）
执行指令后
PC = 地址 （Jump）
TOS = 地址 （HERE + 2）

RESET **复位**

语法：	RESET
操作数：	无
操作：	将所有受 $\overline{MCLR}$ 复位影响的寄存器和标志位复位。
受影响的状态位：	全部
机器码：	0000 0000 1111 1111
说明：	此指令可实现用软件执行 $\overline{MCLR}$ 复位。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	开始复位	空操作	空操作

示例： RESET

执行指令后
寄存器 = 复位值
标志位 * = 复位值

RETFIE 从中断返回

语法： RETFIE {s}

操作数： s ∈ [0,1]

操作： (TOS) → PC，
1 → GIE/GIEH 或 PEIE/GIEL，
如果 s = 1
(WS) → W，
(STATUSS) → STATUS，
(BSRS) → BSR，
PCLATU 和 PCLATH 保持不变。

受影响的状态位： GIE/GIEH 和 PEIE/GIEL

机器码：

0000	0000	0001	000s

说明： 从中断返回。执行出栈操作，将栈顶（TOS）的内容装入 PC。通过将高或低优先级全局中断允许位置 1，来允许中断。如果 s = 1，则影子寄存器 WS、STATUSS 和 BSRS 的内容将被装入对应的寄存器 W、 STATUS 和 BSR。如果 s = 0，则不更新这些寄存器 （默认）。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	从堆栈弹出 PC 值 将 GIEH 或 GIEL 置 1
空操作	空操作	空操作	空操作

示例： RETFIE 1

中断后

PC	=	TOS
W	=	WS
BSR	=	BSRS
STATUS	=	STATUSS
GIE/GIEH， PEIE/GIEL	=	1

RETLW 将立即数返回给 W

语法： RETLW k

操作数： 0 ≤ k ≤ 255

操作： k → W，
(TOS) → PC，
PCLATU 和 PCLATH 保持不变。

受影响的状态位： 无

机器码：

0000	1100	kkkk	kkkk

说明： 将 8 位立即数 k 装入 W。将栈顶内容（返回地址）装入程序计数器。高位地址锁存器 （PCLATH）内容保持不变。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	从堆栈弹出 PC 值， 写入 W
空操作	空操作	空操作	空操作

示例：

```
    CALL TABLE ; W contains table
               ; offset value
               ; W now has
               ; table value
   :
TABLE
    ADDWF PCL  ; W = offset
    RETLW k0   ; Begin table
    RETLW k1   ;
   :
   :
    RETLW kn   ; End of table
```

执行指令前
W = 07h
执行指令后
W = kn 的值

 超前信息

RETURN 从子程序返回

语法：	RETURN {s}
操作数：	s ∈ [0,1]
操作：	(TOS) → PC， 如果 s = 1 (WS) → W， (STATUSS) → STATUS， (BSRS) → BSR， PCLATU 和 PCLATH 保持不变。
受影响的状态位：	无
机器码：	0000 0000 0001 001s
说明：	从子程序返回。执行出栈操作，将栈顶（TOS）内容装入程序计数器。如果 s = 1，将影子寄存器 WS、STATUSS 和 BSRS 的内容装入相应的 W、STATUS 和 BSR 寄存器。如果 s = 0，则不更新这些寄存器（默认）。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	从堆栈弹出 PC 值
空操作	空操作	空操作	空操作

示例： RETURN

执行指令后：
PC = TOS

RLCF f 带进位循环左移

语法：	RLCF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<n>) → dest<n + 1>， (f<7>) → C， (C) → dest<0>
受影响的状态位：	C、N 和 Z
机器码：	0011 01da ffff ffff
说明：	将寄存器 f 的内容连同进位标志位一起循环左移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

C ← 寄存器 f ←

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： RLCF REG, 0, 0

```
执行指令前
    REG  =  1110 0110
    C    =  0
执行指令后
    REG  =  1110 0110
    W    =  1100 1100
    C    =  1
```

RLNCF　　f 循环左移 （不带进位）

语法：	RLNCF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<n>) → dest<n + 1>， (f<7>) → dest<0>
受影响的状态位：	N 和 Z
机器码：	`0100` `01da` `ffff` `ffff`
说明：	将寄存器 f 的内容循环左移 1 位。如果 d 为 `0`，结果存储在 W 中。如果 d 为 `1`，结果存回寄存器 f（默认）。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 `0` 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
	寄存器 f
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例：　`RLNCF　REG, 1, 0`

执行指令前
　REG = `1010 1011`
执行指令后
　REG = `0101 0111`

RRCF　　f 带进位循环右移

语法：	RRCF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<n>) → dest<n – 1>， (f<0>) → C， (C) → dest<7>
受影响的状态位：	C、 N 和 Z
机器码：	`0011` `00da` `ffff` `ffff`
说明：	将寄存器 f 的内容连同进位标志位一起循环右移 1 位。如果 d 为 `0`，结果存储在 W 中。如果 d 为 `1`，结果存回寄存器 f（默认）。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 `0` 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
	C　寄存器 f
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例：　`RRCF　REG, 0, 0`

执行指令前
　REG = `1110 0110`
　C = `0`
执行指令后
　REG = `1110 0110`
　W = `0111 0011`
　C = `0`

超前信息

RRNCF **f 循环右移 （不带进位）**

语法： RRNCF f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： (f<n>) → dest<n – 1>,
(f<0>) → dest<7>

受影响的状态位： N 和 Z

机器码：

0100	00da	ffff	ffff

说明： 将寄存器 f 的内容循环右移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区，忽略 BSR 的值。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

寄存器 f

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1： `RRNCF REG, 1, 0`

执行指令前
REG = 1101 0111
执行指令后
REG = 1110 1011

例 2： `RRNCF REG, 0, 0`

执行指令前
W = ?
REG = 1101 0111
执行指令后
W = 1110 1011
REG = 1101 0111

SETF **将 f 的内容置为全 1**

语法： SETF f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

操作： FFh → f

受影响的状态位： 无

机器码：

0110	100a	ffff	ffff

说明： 将指定寄存器的内容置为 FFh。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `SETF REG,1`

执行指令前
REG = 5Ah
执行指令后
REG = FFh

SLEEP 进入休眠模式

语法： SLEEP

操作数： 无

操作： 00h → WDT，
0 →WDT 后分频器，
1 → $\overline{\text{TO}}$，
0 → $\overline{\text{PD}}$

受影响的状态位： $\overline{\text{TO}}$ 和 $\overline{\text{PD}}$

机器码：

0000	0000	0000	0011

说明： 掉电状态位 （$\overline{\text{PD}}$）清零。超时状态位（$\overline{\text{TO}}$）置 1。看门狗定时器及其后分频器清零。

振荡器停振，处理器进入休眠模式。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	进入 休眠模式

示例： SLEEP

执行指令前
$\overline{\text{TO}}$ = ?
$\overline{\text{PD}}$ = ?

执行指令后
$\overline{\text{TO}}$ = 1 †
$\overline{\text{PD}}$ = 0

† 如果由 WDT 引起唤醒，则此位将被清零。

SUBFWB **W 减去 f （带借位）**

语法： SUBFWB f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： (W) – (f) – ($\overline{\text{C}}$) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0101	01da	ffff	ffff

说明： 将 W 的内容减去 f 寄存器的内容和进位（借位）（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展指令集，只要 $f \le 95$ （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

例 1： SUBFWB REG, 1, 0

执行指令前
REG = 3
W = 2
C = 1

执行指令后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ；结果为负

例 2： SUBFWB REG, 0, 0

执行指令前
REG = 2
W = 5
C = 1

执行指令后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ；结果为正

例 3： SUBFWB REG, 1, 0

执行指令前
REG = 1
W = 2
C = 0

执行指令后
REG = 0
W = 2
C = 1
Z = 1 ；结果为零
N = 0

SUBLW 立即数减去 W 的内容

语法：	SUBLW k
操作数：	0 ≤ k ≤ 255
操作：	k – (W) → W
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0000 1000 kkkk kkkk
说明：	用 8 位立即数 k 减去 W。结果存储在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

例 1： SUBLW 02h

```
执行指令前
    W   =  01h
    C   =  ?
执行指令后
    W   =  01h
    C   =  1    ；结果为正
    Z   =  0
    N   =  0
```

例 2： SUBLW 02h

```
执行指令前
    W   =  02h
    C   =  ?
执行指令后
    W   =  00h
    C   =  1    ；结果为零
    Z   =  1
    N   =  0
```

例 3： SUBLW 02h

```
执行指令前
    W   =  03h
    C   =  ?
执行指令后
    W   =  FFh  ；（二进制补码）
    C   =  0    ；结果为负
    Z   =  0
    N   =  1
```

SUBWF f 减去 W

语法：	SUBWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) – (W) → dest
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0101 11da ffff ffff
说明：	用寄存器 f 中的内容减去 W 寄存器的内容（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1： SUBWF REG, 1, 0

```
执行指令前
    REG  =  3
    W    =  2
    C    =  ?
执行指令后
    REG  =  1
    W    =  2
    C    =  1   ；结果为正
    Z    =  0
    N    =  0
```

例 2： SUBWF REG, 0, 0

```
执行指令前
    REG  =  2
    W    =  2
    C    =  ?
执行指令后
    REG  =  2
    W    =  0
    C    =  1   ；结果为零
    Z    =  1
    N    =  0
```

例 3： SUBWF REG, 1, 0

```
执行指令前
    REG  =  1
    W    =  2
    C    =  ?
执行指令后
    REG  =  FFh ；（二进制补码）
    W    =  2
    C    =  0   ；结果为负
    Z    =  0
    N    =  1
```

SUBWFB **f 减去 W （带借位）**

语法：	SUBWFB f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) – (W) – ($\overline{C}$) → dest
受影响的状态位：	N、 OV、 C、 DC 和 Z
机器码：	`0101` `10da` `ffff` `ffff`
说明：	用 f 寄存器的内容减去 W 的内容和进位（借位）（通过二进制补码方式进行运算）。如果 d 为 `0`，结果存储在 W 中。如果 d 为 `1`，结果存回寄存器 f （默认）。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 `0` 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

例 1： `SUBWFB REG, 1, 0`

执行指令前
```
REG  =  19h   (0001 1001)
W    =  0Dh   (0000 1101)
C    =  1
```
执行指令后
```
REG  =  0Ch   (0000 1011)
W    =  0Dh   (0000 1101)
C    =  1
Z    =  0
N    =  0     ；结果为正
```

例 2： `SUBWFB REG, 0, 0`

执行指令前
```
REG  =  1Bh   (0001 1011)
W    =  1Ah   (0001 1010)
C    =  0
```
执行指令后
```
REG  =  1Bh   (0001 1011)
W    =  00h
C    =  1
Z    =  1     ；结果为零
N    =  0
```

例 3： `SUBWFB REG, 1, 0`

执行指令前
```
REG  =  03h   (0000 0011)
W    =  0Eh   (0000 1101)
C    =  1
```
执行指令后
```
REG  =  F5h   (1111 0100)
              ；[ 二进制补码 ]
W    =  0Eh   (0000 1101)
C    =  0
Z    =  0
N    =  1     ；结果为负
```

SWAPF **将 f 的高半字节和低半字节交换**

语法：	SWAPF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<3:0>) → dest<7:4>， (f<7:4>) → dest<3:0>
受影响的状态位：	无
机器码：	`0011` `10da` `ffff` `ffff`
说明：	f 寄存器的高半字节和低半字节相互交换。如果 d 为 `0`，结果存储在 W 中。如果 d 为 `1`，结果存回寄存器 f （默认）。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 `0` 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例： `SWAPF REG, 1, 0`

执行指令前
```
REG  =  53h
```
执行指令后
```
REG  =  35h
```


TBLRD **表读**

语法： TBLRD (*; *+; *-; +*)

操作数： 无

操作：
如果执行 TBLRD *，
(程序存储器 (TBLPTR))→ TABLAT ；
TBLPTR 不改变
如果执行 TBLRD *+，
(程序存储器 (TBLPTR))→ TABLAT ；
(TBLPTR) + 1 → TBLPTR
如果执行 TBLRD *-，
(程序存储器 (TBLPTR))→ TABLAT ；
(TBLPTR) – 1 → TBLPTR
如果执行 TBLRD +*，
(TBLPTR) + 1 → TBLPTR ；
(程序存储器 (TBLPTR)) → TABLAT

受影响的状态位：无

机器码：

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

说明：
该指令用于读取程序存储器 （P.M.）的内容。使用表指针 （TBLPTR）对程序存储器进行寻址。

TBLPTR （一个 21 位指针）指向程序存储器的每个字节。 TBLPTR 的寻址范围为 2MB。

TBLPTR[0] = 0：程序存储器字的低有效字节

TBLPTR[0] = 1：程序存储器字的高有效字节

TBLRD 指令可用如下方法修改 TBLPTR 的值：

- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作 （读程序存储器）	空操作	空操作 （写 TABLAT）

TBLRD **表读 （续）**

例 1： `TBLRD *+ ;`

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
存储单元 （00A356h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 00A357h

例 2： `TBLRD +* ;`

执行指令前
TABLAT = AAh
TBLPTR = 01A357h
存储单元 （01A357h） = 12h
存储单元 （01A358h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 01A358h

TBLWT 表写

语法： TBLWT (*; *+; *-; +*)

操作数： 无

操作：
如果执行 TBLWT*，
(TABLAT) → 保持寄存器；
TBLPTR 不改变
如果执行 TBLWT*+，
(TABLAT) → 保持寄存器；
(TBLPTR) + 1 → TBLPTR
如果执行 TBLWT*-，
(TABLAT) → 保持寄存器；
(TBLPTR) – 1 → TBLPTR
如果执行 TBLWT+*，
(TBLPTR) + 1 → TBLPTR ；
(TABLAT) → 保持寄存器

受影响的状态位： 无

机器码：

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

说明： 此指令使用 TBLPTR 的低 3 位来确定要将 TABLAT 中的内容写入 8 个保持寄存器中的哪一个。该保持寄存器用于对程序存储器 （P.M.）的内容编程。（关于对闪存程序存储器编程的更多详细信息，请参见**第 5.0 节 “存储器构成”**。）

TBLPTR （一个 21 位指针）指向程序存储器的每个字节。 TBLPTR 的寻址范围为 2MB。 TBLPTR 的 LSb 选择要访问的程序存储器单元。

TBLPTR[0] = 0：程序存储器字的低有效字节

TBLPTR[0] = 1：程序存储器字的高有效字节

TBLWT 指令可用如下方法修改 TBLPTR 的值：

- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作（读 TABLAT）	空操作	空操作（写保持寄存器）

TBLWT 表写 （续）

例 1： `TBLWT *+;`

执行指令前
- TABLAT = 55h
- TBLPTR = 00A356h
- 保持寄存器（00A356h） = FFh

执行指令后 （表写操作完成）
- TABLAT = 55h
- TBLPTR = 00A357h
- 保持寄存器（00A356h） = 55h

例 2： `TBLWT +*;`

执行指令前
- TABLAT = 34h
- TBLPTR = 01389Ah
- 保持寄存器（01389Ah） = FFh
- 保持寄存器（01389Bh） = FFh

执行指令后 （表写操作完成）
- TABLAT = 34h
- TBLPTR = 01389Bh
- 保持寄存器（01389Ah） = FFh
- 保持寄存器（01389Bh） = 34h

TSTFSZ	**测试 f，为 0 则跳过**
语法：	TSTFSZ f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	f 为 0 则跳过
受影响的状态位：	无
机器码：	`0110` `011a` `ffff` `ffff`
说明：	如果 f = 0，丢弃已取的指令并执行一条 NOP 指令，使这条指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1(2) **注：** 如果跳过的指令后面跟有 2 字指令，则执行 TSTFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     TSTFSZ  CNT, 1
NZERO    :
ZERO     :
```

执行指令前
PC = 地址 （HERE）
执行指令后
如果 CNT = 00h，
PC = 地址 （ZERO）
如果 CNT ≠ 00h，
PC = 地址 （NZERO）

XORLW	**将立即数与 W 作逻辑异或运算**
语法：	XORLW k
操作数：	0 ≤ k ≤ 255
操作：	(W) .XOR. k → W
受影响的状态位：	N 和 Z
机器码：	`0000` `1010` `kkkk` `kkkk`
说明：	将 W 的内容与 8 位立即数 k 进行逻辑异或运算。结果存储在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例：

```
XORLW   0AFh
```

执行指令前
W = B5h
执行指令后
W = 1Ah

XORWF 将 W 与 f 作逻辑异或运算

语法：	XORWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(W) .XOR.(f) → dest
受影响的状态位：	N 和 Z
机器码：	`0001` `10da` `ffff` `ffff`
说明：	将 W 的内容与寄存器 f 的内容进行逻辑异或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详细信息，请参见**第 25.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： `XORWF REG, 1, 0`

执行指令前
REG = AFh
W = B5h
执行指令后
REG = 1Ah
W = B5h

25.2 扩展指令集

除了PIC18指令集的75条标准指令之外，PIC18F97J60 系列器件还提供了针对内核CPU功能的可选扩展指令。这些新增的功能包括 8 条额外的指令，它们可以实现间接和变址寻址操作，并使得许多标准 PIC18 指令可以实现立即数变址寻址。

扩展指令集的额外功能在默认情况下对未编程器件是使能的。用户必须在编程期间将 XINST 配置位正确置 1 或清零，从而使能或禁止这些功能。

扩展指令集中的指令可以全部被归为立即数操作类指令，它们既可以控制文件选择寄存器，也可以使用这些寄存器进行变址寻址。其中的两个指令 ADDFSR 和SUBFSR，可以直接对 FSR2 进行操作。而 ADDULNK 和SUBULNK 指令允许在执行后自动返回。

这些扩展的指令专门用于优化用高级语言特别是C语言编写的重入程序代码（也就是递归调用或使用软件堆栈的代码）。此外，它们使用户能更有效地用高级语言对数据结构执行特定的操作。这些操作包括：

- 在进入和退出子程序时对软件堆栈空间进行动态分配和释放
- 功能指针调用
- 对软件堆栈指针进行控制
- 对软件堆栈中的变量进行控制

表 25-3 提供了扩展指令集中的指令汇总。**第 25.2.2 节“扩展指令集”**对这些指令进行了详细说明。第 360 页的表 25-1 提供了标准和扩展的 PIC18 指令集的操作码字段说明。

注：	扩展指令集和立即数变址寻址模式是专为优化用 C 语言编写的应用程序而设计的，用户可能不会在汇编器中直接使用这些指令。对于那些查看编译器生成代码的用户，这些命令的语法可作为参考。

25.2.1 扩展指令的语法

大部分扩展指令都使用变址参数，同时使用一个文件选择寄存器和某一偏移量来指定源寄存器或目标寄存器。当指令的参数作为变址寻址的一部分时，会用方括号（“[ ]”）把它括起来。这时表示此参数用作变址地址或偏移量。如果 MPASM™ 汇编器发现一个变址地址或偏移量没有被括起来，它就会给出出错信息。

当使能扩展指令集时，括号也用于表示面向字节和面向位的指令中的变址参数。这是对指令语法的额外更改。更多详细信息，请参见**第 25.2.3.1 节“标准 PIC18 命令的扩展指令语法”**。

注：	以前，在 PIC18 和早期的指令集中使用方括号来表示可选参数。在此文本和以后的文本中，可选参数将用大括号（“{ }”）表示。

表 25-3： PIC18 指令集的扩展

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位
ADDFSR	f, k	将立即数与 FSR 相加	1	1110	1000	ffkk	kkkk	无
ADDULNK	k	将立即数与 FSR2 相加并返回	2	1110	1000	11kk	kkkk	无
CALLW		使用 WREG 调用子程序	2	0000	0000	0001	0100	无
MOVSF	z_s, f_d	将 z_s（源）移入（第一个字） f_d（目标）移入（第二个字）	2	1110 1111	1011 ffff	0zzz ffff	zzzz ffff	无
MOVSS	z_s, z_d	将 z_s（源）移入（第一个字） z_d（目标）移入（第二个字）	2	1110 1111	1011 xxxx	1zzz xzzz	zzzz zzzz	无
PUSHL	k	将立即数保存到 FSR2，FSR2 减 1	1	1110	1010	kkkk	kkkk	无
SUBFSR	f, k	FSR 减去立即数	1	1110	1001	ffkk	kkkk	无
SUBULNK	k	FSR2 减去立即数并返回	2	1110	1001	11kk	kkkk	无

25.2.2 扩展指令集

ADDFSR **FSR 的内容与立即数相加**

语法： ADDFSR f, k

操作数： $0 \le k \le 63$
$f \in [0, 1, 2]$

操作： FSR(f) + k → FSR(f)

受影响的状态位： 无

机器码：

1110	1000	ffkk	kkkk

说明： 将由 f 指定的 FSR 的内容加上一个 6 位的立即数 k。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 FSR

示例： `ADDFSR 2, 23h`

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 0422h

ADDULNK **FSR2 的内容与立即数相加并返回**

语法： ADDULNK k

操作数： $0 \le k \le 63$

操作： FSR2 + k → FSR2，
(TOS) → PC

受影响的状态位： 无

机器码：

1110	1000	11kk	kkkk

说明： 将 FSR2 的内容加上一个 6 位的立即数 k。然后通过将 TOS 的值装入 PC，执行一条 RETURN 指令。

执行该指令需要两个周期；在第二个周期执行一条 NOP 指令。

该指令可以被认为是 ADDFSR 指令的特例，其中 f = 3 （二进制 “11”），它仅针对 FSR2 进行操作。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 FSR
空操作	空操作	空操作	空操作

示例： `ADDULNK 23h`

执行指令前
FSR2 = 03FFh
PC = 0100h

执行指令后
FSR2 = 0422h
PC = (TOS)

注：	所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令语法将变为：{label} 指令参数。

CALLW 使用 WREG 调用子程序

语法：	CALLW
操作数：	无
操作：	(PC + 2) → TOS， (W) → PCL， (PCLATH) → PCH， (PCLATU) → PCU
受影响的状态位：	无
机器码：	0000 0000 0001 0100
说明	首先，返回地址 （PC + 2）被压入返回堆栈。接下来，将 W 寄存器的内容写入 PCL， PCL 现有的值被丢弃。然后，PCLATH 和 PCLATU 的内容被分别锁存到 PCH 和 PCU。第二个周期执行一条 NOP 指令，并同时取下一条指令。 和 CALL 不一样，该指令没有更新 W、STATUS 或 BSR 寄存器的选项。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 WREG	将 PC 压入堆栈	空操作
空操作	空操作	空操作	空操作

示例： HERE CALLW

执行指令前
PC = 地址 （HERE）
PCLATH = 10h
PCLATU = 00h
W = 06h

执行指令后
PC = 001006h
TOS = 地址 （HERE + 2）
PCLATH = 10h
PCLATU = 00h
W = 06h

MOVSF 将变址寻址单元内容移入 f

语法：	MOVSF [z_s], f_d
操作数：	$0 \le z_s \le 127$ $0 \le f_d \le 4095$
操作：	((FSR2) + z_s) → f_d
受影响的状态位：	无
机器码： 第一个字 （源） 第二个字 （目标）	1110 1011 0zzz $zzzz_s$ 1111 ffff ffff $ffff_d$
说明：	将源寄存器的内容移入目标寄存器 f_d。通过将第一个字中的 7 位立即数偏移量 z_s 与 FSR2 的值相加来确定源寄存器的实际地址。第二个字中的 12 位立即数 f_d 指向目标寄存器的地址。两个地址均可以是 4096 字节的数据空间 （000h 到 FFFh）中的任何单元。 MOVSF 指令中的目标寄存器不能是 PCL、 TOSU、 TOSH 或 TOSL。 如果计算得到的源地址指向间接寻址寄存器，将返回 00h。
指令字数：	2
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	空操作 无效读取	空操作	写寄存器 f （目标寄存器）

示例： MOVSF [05h], REG2

执行指令前
FSR2 = 80h
85h 单元的内容 = 33h
REG2 = 11h

执行指令后
FSR2 = 80h
85h 单元的内容 = 33h
REG2 = 33h

MOVSS 变址寻址移动数据

语法：	MOVSS [z_s], [z_d]
操作数：	$0 \le z_s \le 127$ $0 \le z_d \le 127$
操作：	$((FSR2) + z_s) \rightarrow ((FSR2) + z_d)$
受影响的状态位：	无

机器码：

第一个字 （源）	1110	1011	1zzz	$zzzz_s$
第二个字 （目标）	1111	xxxx	xzzz	$zzzz_d$

说明

将源寄存器的内容移到目标寄存器。通过将 FSR2 中的值分别加上 7 位立即数偏移量 z_s 和 z_d 来确定源寄存器和目标寄存器的地址。两个寄存器都可以是 4096 字节数据存储器空间 （000h 到 FFFh）中的任意单元。

MOVSS 指令不能使用 PCL、 TOSU、 TOSH 或 TOSL 作为目标寄存器。

如果计算得到的源地址指向间接寻址寄存器，将返回 00h。如果计算得到的目标地址指向间接寻址寄存器，将执行一条 NOP 指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	确定 目标地址	确定 目标地址	写 目标寄存器

示例： `MOVSS [05h], [06h]`

执行指令前
- FSR2 = 80h
- 85h 单元的内容 = 33h
- 86h 单元的内容 = 11h

执行指令后
- FSR2 = 80h
- 85h 单元的内容= 33h
- 86h 单元的内容 = 33h

PUSHL 将立即数保存到 FSR2， FSR2 减 1

语法：	PUSHL k
操作数：	$0 \le k \le 255$
操作：	$k \rightarrow (FSR2)$, $FSR2 - 1 \rightarrow FSR2$
受影响的状态位：	无

机器码：

1111	1010	kkkk	kkkk

说明：

8 位立即数 k 被写入由 FSR2 指定的数据存储器单元。
操作完后 FSR2 减 1。

此指令允许用户将值压入软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入 目标寄存器

示例： `PUSHL 08h`

执行指令前
- FSR2H:FSR2L = 01ECh
- 存储单元 （01ECh） = 00h

执行指令后
- FSR2H:FSR2L = 01EBh
- 存储单元 （01ECh） = 08h

SUBFSR **FSR 减去立即数**

语法：	SUBFSR f, k
操作数：	0 ≤ k ≤ 63 f ∈ [0, 1, 2]
操作：	FSRf – k → FSRf
受影响的状态位：	无
机器码：	`1110` `1001` `ffkk` `kkkk`
说明：	用 f 指定的 FSR 的内容减去 6 位立即数 k。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： `SUBFSR 2, 23h`

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 03DCh

SUBULNK **FSR2 减去立即数并返回**

语法：	SUBULNK k
操作数：	0 ≤ k ≤ 63
操作：	FSR2 – k → FSR2 (TOS) → PC
受影响的状态位：	无
机器码：	`1110` `1001` `11kk` `kkkk`
说明：	用 FSR 的内容减去 6 位立即数 k，然后通过将 TOS 的值装入 PC，执行一条 `RETURN` 指令。 执行该指令需要两个指令周期，第二个指令周期执行一条 `NOP` 指令。 该指令可以被看作是 `SUBFSR` 指令的特殊情况，其中 f = 3（二进制数 “`11`”）；它只针对 FSR2 进行操作。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器
空操作	空操作	空操作	空操作

示例： `SUBULNK 23h`

执行指令前
FSR2 = 03FFh
PC = 0100h

执行指令后
FSR2 = 03DCh
PC = (TOS)

25.2.3 立即数变址寻址模式中面向字节和面向位的指令

注：	使能 PIC18 扩展指令集可能导致常规应用程序运行不正常或完全失败。

一旦使能扩展指令集，除了可以使用 8 条新命令之外，还可以使用立即数变址寻址（**第 5.6.1 节“使用立即数偏移量进行变址寻址”**）。这将导致标准 PIC18 指令集中大部分指令的地址解析方法有很大变化。

当禁止扩展指令集时，嵌入在操作码中的地址被视作立即数存储单元：可以是快速操作存储区中的单元（a = 0），或由 BSR 指定的 GPR 存储区中的单元（a = 1）。当使能扩展指令集且 a = 0 时，地址为 5Fh 或以下的文件寄存器参数被解析为 FSR2 中的指针值的偏移量，而不是一个立即数地址。对于实际应用来说，这意味着所有使用快速操作 RAM 位作为参数的指令，即所有面向字节或面向位的指令，或者几乎半数的 PIC18 内核指令，在使能了扩展指令集时操作都会有所不同。

当 FSR2 的内容为 00h 时，快速操作 RAM 的边界会被重新映射到它们的原始值。这对于编写向下兼容的代码很有用处。如果使用此技术，有必要在“C”程序调用汇编子程序时保存 FSR2 的值并在返回时将它恢复，这样做的目的是保护堆栈指针。用户还必须记住扩展指令集的语法要求（见**第 25.2.3.1 节“标准 PIC18 命令的扩展指令语法”**）。

虽然立即数变址寻址模式对于动态堆栈和指针控制很有用处，但是如果不小心误用了寄存器也会非常麻烦。已经习惯使用 PIC18 编程的用户必须记住，在使能了扩展指令集后，地址小于或等于 5Fh 的寄存器用于立即数变址寻址。

下面是在立即数变址寻址模式中，一些面向字节和位的指令的示例，通过示例可以看出指令执行如何受到影响。示例中的操作数条件适用于所有这一类的指令。

25.2.3.1 标准 PIC18 命令的扩展指令语法

当使能了扩展指令集时，立即数偏移量“k”被用来替换标准的面向字节和位的命令中的文件寄存器参数“f”。如前所述，只有在“f”小于或等于 5Fh 时才会发生这种情况。当使用偏移量时，该偏移量必须用方括号“[]”标出。因为在扩展指令集中，编译器将括号中的数值解析为变址地址或偏移量。省略括号，或在括号内使用大于 5Fh 的值会在 MPASM 汇编器中产生错误。

如果变址参数已被加上了括号，那么就不再需要指定快速操作 RAM 参数；此参数被假定为 0。这与标准操作（禁止扩展指令集时）刚好相反。在变址寻址模式中，声明快速操作 RAM 位也将在 MPASM 汇编器中产生错误。

目标参数“d”的操作和以前一样。

在 MPASM 汇编器的最新版本中，必须明确调用对扩展指令集的语言支持。可以通过命令行选项 /y 或在源代码中加入 PE 伪指令进行调用。

25.2.4 使能扩展指令集时的注意事项

需要注意的是并非所有用户都有必要使用扩展指令集，尤其是那些不使用软件堆栈的用户。

此外，立即数变址寻址模式可能会给写入 PIC18 汇编器的常规应用程序带来问题。这是因为常规的指令会尝试寻址快速操作存储区中地址低于 5Fh 的寄存器。当使能了扩展指令集时，这些地址被解析为相对于 FSR2 的立即数偏移量，所以应用程序会读或写错误的地址。

将应用程序移植到 PIC18F97J60 系列器件时，代码的类型是非常重要的。在使用扩展指令集时，用 C 语言编写的代码较长的重入应用程序会运行地很好，而大量使用快速操作存储区的常规应用程序不会获得任何益处。

ADDWF **将 W 与变址寻址单元的内容相加（立即数变址寻址模式）**

语法：	ADDWF [k] {,d}
操作数：	0 ≤ k ≤ 95 d ∈ [0,1]
操作：	(W) + ((FSR2) + k) → dest
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0010 | 01d0 | kkkk | kkkk
说明：	将 W 的内容与由 FSR2 加上偏移量 k 指定的寄存器的内容相加。 如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入目标寄存器

示例： ADDWF [OFST] ,0

执行指令前
W = 17h
OFST = 2Ch
FSR2 = 0A00h
0A2Ch 单元的内容 = 20h
执行指令后
W = 37h
0A2Ch 单元的内容 = 20h

BSF **将变址寻址单元相应位置 1（立即数变址寻址模式）**

语法：	BSF [k], b
操作数：	0 ≤ f ≤ 95 0 ≤ b ≤ 7
操作：	1 → ((FSR2) + k)<b>
受影响的状态位：	无
机器码：	1000 | bbb0 | kkkk | kkkk
说明：	将由 FSR2 加上偏移量 k 指定的寄存器中的位 b 置 1。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： BSF [FLAG_OFST], 7

执行指令前
FLAG_OFST = 0Ah
FSR2 = 0A00h
0A0Ah 单元的内容 = 55h
执行指令后
0A0Ah 单元的内容 = D5h

SETF **将变址寻址单元置全 1（立即数变址寻址模式）**

语法：	SETF [k]
操作数：	0 ≤ k ≤ 95
操作：	FFh → ((FSR2) + k)
受影响的状态位：	无
机器码：	0110 | 1000 | kkkk | kkkk
说明：	将由 FSR2 加上偏移量 k 指定的寄存器的内容置为 FFh。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写寄存器

示例： SETF [OFST]

执行指令前
OFST = 2Ch
FSR2 = 0A00h
0A2Ch 单元的内容= 00h
执行指令后
0A2Ch 单元的内容= FFh

25.2.5 使用 MICROCHIP MPLAB® IDE 工具的注意事项

最新版本的 Microchip 软件工具完全支持 PIC18F97J60 系列器件的扩展指令集。包括 MPLAB C18 C 语言编译器、MPASM 汇编语言和 MPLAB 集成开发环境（IDE）。

在选择了使用软件开发的目标器件后， MPLAB IDE 将自动按默认模式设置该器件的配置位。XINST 配置位的默认设置是 0，禁用扩展指令集和立即数变址寻址模式。在编程过程中必须将 XINST 位置 1 才能确保使用扩展指令集开发的应用程序能够正确执行。

要使用扩展指令集开发软件，用户必须设置他们的语言工具以实现对扩展指令和变址寻址模式的支持。根据所使用的环境，可以通过以下几种方法：

- 开发环境中的菜单选项或对话框，允许用户配置项目的语言工具及其设置
- 命令行选项
- 源代码中的伪指令

这些选项在不同的编译器、汇编器和开发环境中将有所不同。建议用户在其开发系统所附带的文档中查询相应的信息。

26.0 开发支持

一系列硬件及软件开发工具对 PIC® 单片机提供支持：

- 集成开发环境
  - MPLAB® IDE 软件
- 汇编器 / 编译器 / 链接器
  - MPASM™ 汇编器
  - MPLAB C18 和 MPLAB C30 C 编译器
  - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
  - MPLAB ASM30 汇编器 / 链接器 / 库
- 模拟器
  - MPLAB SIM 软件模拟器
- 仿真器
  - MPLAB ICE 2000 在线仿真器
  - MPLAB REAL ICE™ 在线仿真器
- 在线调试器
  - MPLAB ICD 2
- 器件编程器
  - PICSTART® Plus 开发编程器
  - MPLAB PM3 器件编程器
  - PICkit™ 2 开发编程器
- 低成本演示和开发板及评估工具包

26.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16 位单片机市场提供了前所未有的易于使用的软件开发平台。MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
  - 模拟器
  - 编程器 （单独销售）
  - 仿真器 （单独销售）
  - 在线调试器 （单独销售）
- 具有彩色上下文代码显示的全功能编辑器
- 多项目管理器
- 内容可直接编辑的可定制式数据窗口
- 高级源代码调试
- 可视化器件初始化程序，便于进行寄存器的初始化
- 鼠标停留在变量上进行查看的功能
- 通过拖放把变量从源代码窗口拉到观察窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 HI-TECH 软件 C 编译器和 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件 （汇编语言或 C 语言）
- 点击一次即可完成汇编 （或编译）并将代码下载到 PIC MCU 仿真器和模拟器工具中 （自动更新所有项目信息）
- 可使用如下各项进行调试：
  - 源文件 （汇编语言或 C 语言）
  - 混合汇编语言和 C 语言
  - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能更强大的工具时的学习时间。

26.2 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于所有的 PIC MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特征：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

26.3 MPLAB C18 和 MPLAB C30 C 编译器

MPLAB C18 和 MPLAB C30 代码开发系统是完全的 ANSI C 编译器，分别适用于 Microchip 的 PIC18 系列单片机及 dsPIC30F、dsPIC33 和 PIC24 系列数字信号控制器。这些编译器可提供其他编译器并不具备的强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供了针对 MPLAB IDE 调试器的优化符号信息。

26.4 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用中。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特征：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起来增强代码的可维护性
- 只要列出、替换、删除和抽取模块，便可灵活地创建库

26.5 MPLAB ASM30 汇编器、链接器和库管理器

MPLAB ASM30 汇编器为 dsPIC30F 器件提供转换自符号汇编语言的可重定位机器码。 MPLAB C30 C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特征：

- 支持整个 dsPIC30F 指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

26.6 MPLAB SIM 软件模拟器

MPLAB SIM软件模拟器在指令级对PIC MCU和dsPIC® DSC 进行模拟，使得用户可以在 PC 主机的环境下进行代码开发。对于任何给定的指令，用户均可对数据区进行检查或修改，并通过各种触发机制来产生激励。可以将各寄存器的情况记录在文件中，以便进行进一步地运行时分析。跟踪缓冲器和逻辑分析器的显示使模拟器还能记录和跟踪程序的执行、 I/O 的动作、大部分的外设及内部寄存器的状况。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C18 和 MPLAB C30 C 编译器以及MPASM和MAPLAB ASM30 汇编器的符号调试。该软件模拟器可用于在硬件实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

26.7 MPLAB ICE 2000 高性能在线仿真器

MPLAB ICE 2000 在线仿真器旨在为产品开发工程师提供一整套用于 PIC 单片机的设计工具。 MPLAB ICE 2000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 2000 是全功能仿真器系统，它具有增强的跟踪、触发和数据监控功能。处理器模块可插拔，使系统可轻松进行重新配置以适应各种不同处理器的仿真需要。MPLAB ICE 2000 在线仿真器的架构允许对其进行扩展以支持新的 PIC 单片机。

MPLAB ICE 2000 在线仿真器系统设计为一款实时仿真系统，该仿真系统具备通常只有昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft® Windows® 32 位操作系统可使这些功能在一个简单而统一的应用中得到很好的利用。

26.8 MPLAB REAL ICE 在线仿真器系统

MPLAB REAL ICE在线仿真器系统是Microchip针对其闪存 DSC® 和 MCU 器件而推出的新一代高速仿真器。结合 MPLAB 集成开发环境（IDE）所具有的易于使用且功能强大的图形用户界面，该仿真器可对 PIC® 和 dsPIC® 闪存单片机进行调试和编程。IDE 是随每个工具包一起提供的。

MPLAB REAL ICE 探针通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与常用 MPLAB ICD 2 系统兼容的连接器（RJ11）或新型抗噪声、高速低压差分信号（LVDS）互连电缆（CAT5）与目标板相连。

可通过 MPLAB IDE 下载将来版本的固件，对 MPLAB REAL ICE 进行现场升级。在即将推出的 MPLAB IDE 版本中，会支持许多新器件，还将增加一些新特性，如软件断点和汇编代码跟踪等。在同类仿真器中， MPLAB REAL ICE 的优势十分明显：低成本、高速仿真、实时变量监视、跟踪分析、复杂断点、耐用的探针接口及较长（长达 3 米）的互连电缆。

26.9 MPLAB ICD 2 在线调试器

Microchip 的在线调试器 MPLAB ICD 2 是一款功能强大而成本低廉的运行时开发工具，通过 RS-232 或高速 USB 接口与 PC 主机相连。该工具基于闪存 PIC MCU，可用于开发本系列及其他 PIC MCU 和 dsPIC DSC。MPLAB ICD 2 使用了闪存器件中内建的在线调试功能。该功能结合 Microchip 的在线串行编程（In-Circuit Serial Programming™，ICSP™）协议，可在 MPLAB 集成开发环境的图形用户界面上提供成本效益很高的在线闪存调试。这使设计人员可通过设置断点、单步运行以及对变量、CPU 状态以及外设寄存器进行监视的方法实现源代码的开发和调试。其全速运行特性可对硬件和应用进行实时测试。 MPLAB ICD 2 还可用作某些 PIC 器件的开发编程器。

26.10 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款通用的、符合 CE 规范的器件编程器，其可编程电压设置在 V_{DDMIN} 和 V_{DDMAX} 之间时可靠性最高。它有一个用来显示菜单和错误信息的大 LCD 显示器（128 x 64），以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP™ 电缆。在单机模式下， MPLAB PM3 器件编程器不必与 PC 相连即可对 PIC 器件进行读取、验证和编程。在该模式下它还可设置代码保护。 MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。MPLAB PM3 具备高速通信能力以及优化算法，可对存储器很大的器件进行快速编程，它还采用 SD/MMC 卡用作文件存储及数据安全应用。

26.11 PICSTART Plus 开发编程器

PICSTART Plus开发编程器是一款易于使用而成本低廉的原型编程器。它通过 COM （RS-232）端口与 PC 相连。 MPLAB 集成开发环境软件使得该编程器的使用简便、高效。PICSTART Plus 开发编程器支持采用 DIP 封装的大部分 PIC 器件，其引脚数最多可达 40 个。引脚数更多的器件，如 PIC16C92X 和 PIC17C76X，可通过连接一个转接插槽来获得支持。PICSTART Plus 开发编程器符合 CE 规范。

26.12 PICkit 2 开发编程器

PICkit™ 2 开发编程器是一个低成本编程器；对于某些选定闪存器件，它也是一个调试器，通过其易于使用的接口可对众多 Microchip 的低档、中档和 PIC18F 系列闪存单片机进行编程。 PICkit 2 入门工具包中包含一个有实验布线区的开发板、十二堂系列课程、软件和 HI-TECH 的 PICC™ Lite C 编译器，有助于用户快速掌握 PIC® 单片机的使用。这一工具包为使用 Microchip 功能强大的中档闪存系列单片机进行编程、评估和应用开发，提供了所需的一切。

26.13 演示、开发和评估板

有许多演示、开发和评估板可用于各种 PIC MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于测试和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、 RS-232 接口、 LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外， Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、 KEELOQ® 数据安全产品 IC、CAN、 IrDA®、 PowerSmart® 电池管理、 SEEVAL® 评估系统、 Σ–Δ ADC、流速传感器，等等。

有关演示、开发和评估工具包的完整列表，请查阅 Microchip 公司网页 （www.microchip.com）以及最新的 *“Product Selector Guide（产品选型指南）”*（DS00148）。

27.0 电气特性

绝对极限参数值 (†)

偏置电压下的环境温度 .... -40°C 至 +100°C
储存温度 .... -65°C 至 +150°C
任一仅数字输入引脚或 $\overline{MCLR}$ 相对于 V_{SS} 的电压（除 V_{DD} 外）........ -0.3V 至 6.0V
任一数模组合引脚相对于 V_{SS} 的电压 .... -0.3V 至（V_{DD} + 0.3V）
V_{DDCORE} 引脚相对于 V_{SS} 的电压 .... -0.3V 至 2.75V
V_{DD} 引脚相对于 V_{SS} 的电压 .... -0.3V 至 3.6V
总功耗（注 1）........ 1.0W
流出 V_{SS} 引脚的最大电流 .... 300 mA
流入 V_{DD} 引脚的最大电流 .... 250 mA
输入箝位电流，I_{IK}（$V_I < 0$ 或 $V_I > V_{DD}$）（注 3）........ ±0 mA
输出箝位电流，I_{OK}（$V_O < 0$ 或 $V_O > V_{DD}$）（注 3）........ ±0 mA
PORTB 和 PORTC I/O 引脚的最大输出灌电流 .... 25 mA
PORTD、 PORTE 和 PORTJ I/O 引脚的最大输出灌电流 .... 8 mA
PORTA[(2)]、 PORTF、 PORTG 和 PORTH I/O 引脚的最大输出灌电流 .... 2 mA
PORTB 和 PORTC I/O 引脚的最大输出拉电流 .... 25 mA
PORTD、 PORTE 和 PORTJ I/O 引脚的最大输出拉电流 .... 8 mA
PORTA[(2)]、 PORTF、 PORTG 和 PORTH I/O 引脚的最大输出拉电流 .... 2 mA
所有组合端口的最大灌电流 .... 200 mA
所有组合端口的最大拉电流 .... 200 mA

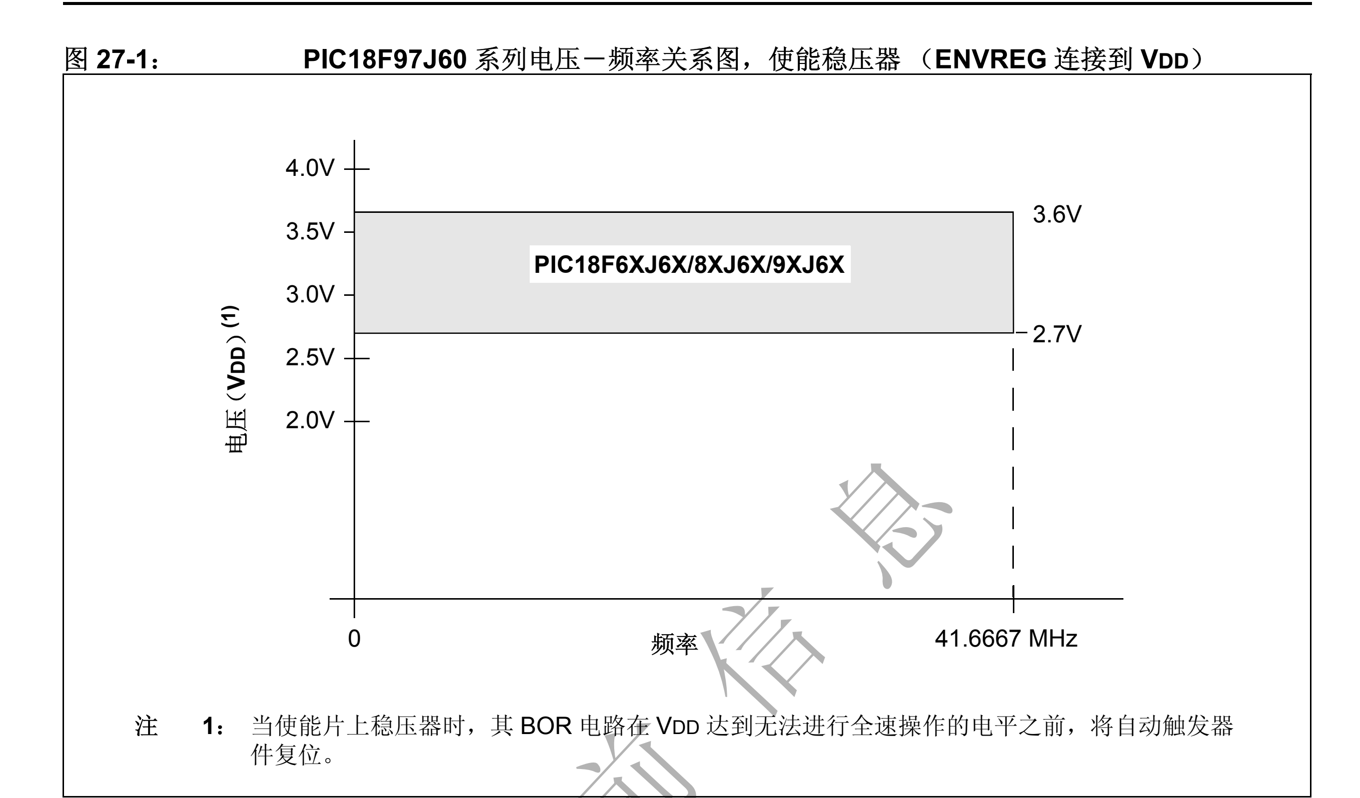
注 1：功耗计算公式为：
$Pdis = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL}) + \sum (V_{TPOUT} \times I_{TPOUT})$

2：例外情况是 RA0:RA1，可高达 8 mA，能够直接驱动 LED。

3：不存在箝位二极管。

† 注：如果运行条件超过了上述 “绝对极限参数值”，即可能对器件造成永久性损坏。上述值仅为运行条件的极大值，我们不建议器件运行在该规范范围以外。器件长时间工作在绝对极限参数条件下，其稳定性可能受到影响。

图 27-1： PIC18F97J60 系列电压－频率关系图，使能稳压器 （ENVREG 连接到 VDD）



注 1： 当使能片上稳压器时，其 BOR 电路在 VDD 达到无法进行全速操作的电平之前，将自动触发器件复位。

图 27-2： PIC18F97J60 系列电压－频率关系图，禁止稳压器 （ENVREG 连接到 VSS）

3.00V
2.75V
2.50V
2.25V
2.00V
电压（VDDCORE）(1)
PIC18F6XJ6X/8XJ6X/9XJ6X
2.7V
2.35V
4 MHz
41.6667 MHz
频率

对于 4 MHz 到 41.6667 MHz 之间的频率， $F_{MAX} = (107.619\ MHz/V) * (V_{DDCORE} - 2V) + 4\ MHz$

注 1： 当禁止片上稳压器时， VDD 和 VDDCORE 必须维持在 $V_{DDCORE} \le V_{DD} \le 3.6V$。

27.1 直流特性： 供电电压， PIC18F97J60 系列 （工业级）

PIC18F97J60 系列（工业级）			标准运行条件（除非另外说明） 运行温度 工业级为 -40°C ≤ TA ≤ +85°C				
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
D001	VDD	供电电压	VDDCORE 2.65	— —	3.6 3.6	V V	ENVREG = 0 ENVREG = 1
D001B	VDDCORE	单片机内核的外部电源	2.00	—	2.70	V	
D001C	AVDD	模拟供电电压	VDD – 0.3	—	VDD + 0.3	V	
D002	VDR	**RAM 数据保持电压** [(1)]	1.5	—	—	V	
D003	VPOR	**VDD** 上电复位电压	—	—	TBD	V	详见第 **4.3** 节 “上电复位（**POR**）”。
D004	SVDD	**VDD** 上升率 （确保内部上电复位）	0.05	—	—	V/ms	详见第 **4.3** 节 “上电复位（**POR**）”。

注 1： 这是在不丢失 RAM 数据的前提下，休眠模式或器件复位期间 VDD 所能降到的最小电压值。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级）

PIC18F97J60 系列 （工业级）		标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ T_A ≤ +85°C				
参数编号	器件	典型值	最大值	单位	条件	
	掉电电流 （I_{PD}）[(1)]					
	所有器件	TBD	TBD	μA	-40°C	V_{DD} = 2.0V， V_{DDCORE} = 2.0V， ENVREG = 0 （休眠模式）
		TBD	TBD	μA	+25°C	
		TBD	TBD	μA	+85°C	
	所有器件	TBD	TBD	μA	-40°C	V_{DD} = 2.5V， V_{DDCORE} = 2.5V， ENVREG = 0 （休眠模式）
		TBD	TBD	μA	+25°C	
		TBD	TBD	μA	+85°C	
	所有器件	TBD	TBD	μA	-40°C	V_{DD} = 3.3V， ENVREG = 1 （休眠模式）
		TBD	TBD	μA	+25°C	
		TBD	TBD	μA	+85°C	

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 V_{DD} 或 V_{SS}，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 I_{DD} 测量值的测试条件为：
$\overline{\text{OSC1}}$ = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 V_{DD} ；
MCLR = V_{DD} ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级） （续）

PIC18F97J60 系列 （工业级）		标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ TA ≤ +85°C					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD） (2,3)						
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.0V，VDDCORE = 2.0V，ENVREG = 0	FOSC = 31 kHz （**RC_RUN** 模式，内部振荡器作为时钟源）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.5V，VDDCORE = 2.5V，ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 3.3V，ENVREG = 1	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.0V，VDDCORE = 2.0V，ENVREG = 0	FOSC = 31 kHz （**RC_IDLE** 模式，内部振荡器作为时钟源）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.5V，VDDCORE = 2.5V，ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 3.3V，ENVREG = 1	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 IDD 测量值的测试条件为：
OSC1 = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 VDD ；
MCLR = VDD ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级） （续）

PIC18F97J60 系列（工业级）		标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ T_A ≤ +85°C					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （I_{DD}）(2,3)						
	所有器件	TBD	TBD	μA	-40°C	V_{DD} = 2.0V，V_{DDCORE} = 2.0V，ENVREG = 0	F_{OSC} = 1 MHz（**PRI_RUN** 模式，EC 振荡器）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	V_{DD} = 2.5V，V_{DDCORE} = 2.5V，ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	V_{DD} = 3.3V，ENVREG = 1	
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	V_{DD} = 2.5V，V_{DDCORE} = 2.5V，ENVREG = 0	F_{OSC} = 25 MHz（**PRI_RUN** 模式，EC 振荡器）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	V_{DD} = 3.3V，ENVREG = 1	
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	V_{DD} = 2.5V，V_{DDCORE} = 2.5V，ENVREG = 0	F_{OSC} = 41.6667 MHz（**PRI_RUN** 模式，EC 振荡器）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	V_{DD} = 3.3V，ENVREG = 1	
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 V_{DD} 或 V_{SS}，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 I_{DD} 测量值的测试条件为：
OSC1 = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 V_{DD} ；
MCLR = V_{DD} ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级） （续）

PIC18F97J60 系列 （工业级）		标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ TA ≤ +85°C					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）[2]						
	所有器件	TBD	TBD	mA	-40°C	VDD = 2.5V，VDDCORE = 2.5V，ENVREG = 0	FOSC = 25 MHz，2.7778 MHz 内部（PRI_RUN HS 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 3.3V，ENVREG = 1	FOSC = 25 MHz，2.7778 MHz 内部（**PRI_RUN HS** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 2.5V，VDDCORE = 2.5V，ENVREG = 0	FOSC = 25 MHz，13.8889 MHz 内部（**PRI_RUN HSPLL** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 3.3V，ENVREG = 1	FOSC = 25 MHz，13.8889 MHz 内部（**PRI_RUN HSPLL** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 2.5V，VDDCORE = 2.5V，ENVREG = 0	FOSC = 25 MHz，25 MHz 内部（**PRI_RUN HS** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 3.3V，ENVREG = 1	FOSC = 25 MHz，25 MHz 内部（**PRI_RUN HS** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 2.5V，VDDCORE = 2.5V，ENVREG = 0	FOSC = 25 MHz，41.6667 MHz 内部（**PRI_RUN HSPLL** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 3.3V，ENVREG = 1	FOSC = 25 MHz，41.6667 MHz 内部（**PRI_RUN HSPLL** 模式）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 IDD 测量值的测试条件为：
OSC1 = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 VDD ；
MCLR = VDD ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级） （续）

PIC18F97J60 系列 （工业级）		标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ TA ≤ +85°C					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD） (2,3)						
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.0V， VDDCORE = 2.0V， ENVREG = 0	FOSC = 1 MHz （**PRI_IDLE** 模式， EC 振荡器）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.5V， VDDCORE = 2.5V， ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 3.3V， ENVREG = 1	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	μA	-40°C	VDD = 2.5V， VDDCORE = 2.5V， ENVREG = 0	FOSC = 25 MHz （**PRI_IDLE** 模式， EC 振荡器）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 3.3V， ENVREG = 1	
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 2.5V， VDDCORE = 2.5V， ENVREG = 0	FOSC = 41.6667 MHz （**PRI_IDLE** 模式， EC 振荡器）
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		
	所有器件	TBD	TBD	mA	-40°C	VDD = 3.3V， ENVREG = 1	
		TBD	TBD	mA	+25°C		
		TBD	TBD	mA	+85°C		

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 IDD 测量值的测试条件为：
OSC1 = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 VDD ；
MCLR = VDD ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级）（续）

PIC18F97J60 系列（工业级）		标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ TA ≤ +85°C					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）(2,3)						
	所有器件	TBD	TBD	μA	-10°C	VDD = 2.0V， VDDCORE = 2.0V， ENVREG = 0	FOSC = 32 kHz(4) （SEC_RUN 模式， Timer1 作为时钟源）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+70°C		
	所有器件	TBD	TBD	μA	-10°C	VDD = 2.5V， VDDCORE = 2.5V， ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+70°C		
	所有器件	TBD	TBD	μA	-10°C	VDD = 3.3V， ENVREG = 1	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+70°C		
	所有器件	TBD	TBD	μA	-10°C	VDD = 2.0V， VDDCORE = 2.0V， ENVREG = 0	FOSC = 32 kHz(4) （SEC_IDLE 模式， Timer1 作为时钟源）
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+70°C		
	所有器件	TBD	TBD	μA	-10°C	VDD = 2.5V， VDDCORE = 2.5V， ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+70°C		
	所有器件	TBD	TBD	μA	-10°C	VDD = 3.3V， ENVREG = 1	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+70°C		

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 IDD 测量值的测试条件为：
OSC1 = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 VDD ；
MCLR = VDD ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.2 直流特性： 掉电和供电电流 PIC18F97J60 系列 （工业级） （续）

PIC18F97J60 系列 （工业级）	标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ T_A ≤ +85°C						
参数编号	器件	典型值	最大值	单位	条件		
	模块差分电流 （ΔI_{WDT}、 ΔI_{OSCB}、 ΔI_{AD} 和 ΔI_{ETH})						
D022 （ΔI_{WDT}）	看门狗定时器	TBD	TBD	μA	-40°C	V_{DD} = 2.0V， V_{DDCORE} = 2.0V， ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
		TBD	TBD	μA	-40°C	V_{DD} = 2.5V， V_{DDCORE} = 2.5V， ENVREG = 0	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
		TBD	TBD	μA	-40°C	V_{DD} = 3.3V， ENVREG = 1	
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
D025 （ΔI_{OSCB}）	Timer1 振荡器	TBD	TBD	μA	-40°C	V_{DD} = 2.0V， V_{DDCORE} = 2.0V， ENVREG = 0	Timer1[3] 为 32 kHz
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
		TBD	TBD	μA	-40°C	V_{DD} = 2.5V， V_{DDCORE} = 2.5V， ENVREG = 0	Timer1[3] 为 32 kHz
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
		TBD	TBD	μA	-40°C	V_{DD} = 3.3V， ENVREG = 1	Timer1[3] 为 32 kHz
		TBD	TBD	μA	+25°C		
		TBD	TBD	μA	+85°C		
D026 （ΔI_{AD}）	A/D 转换器	TBD	TBD	μA	-40°C 至 +85°C	V_{DD} = 2.0V， V_{DDCORE} = 2.0V， ENVREG = 0	A/D 启动，但不进行转换
		TBD	TBD	μA	-40°C 至 +85°C	V_{DD} = 2.5V， V_{DDCORE} = 2.5V， ENVREG = 0	
		TBD	TBD	μA	-40°C 至 +85°C	V_{DD} = 3.3V， ENVREG = 1	
D027 ΔI_{ETH}	以太网模块	140	TBD	mA	-40°C 至 +85°C	V_{DD} = 3.3V， ENVREG = 1	包括 TPOUT+ 和 TPOUT- 的灌电流， LEDA 和 LEDB 被禁止

图注： TBD = 待定

注 1： 在休眠模式下，掉电电流并不取决于振荡器的类型。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 V_{DD} 或 V_{SS}，禁止所有会带来新增电流的功能部件 （如 WDT、 Timer1 振荡器等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
上电工作模式下，所有 I_{DD} 测量值的测试条件为：
OSC1 = 外部方波，轨到轨摆幅；所有 I/O 引脚均为三态，上拉至 V_{DD} ；
$\overline{MCLR}$ = V_{DD} ；根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展温度晶振成本则高很多。

27.3 直流特性： PIC18F97J60 系列 （工业级）

直流特性			标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ T_A ≤ +85°C			
参数编号	符号	特性	最小值	最大值	单位	条件
	V_{IL}	输入低电压				
		I/O 端口：				
D030		带 TTL 缓冲器	V_{SS}	0.15 V_{DD}	V	
D031		带施密特触发缓冲器	V_{SS}	0.2 V_{DD}	V	
D032		$\overline{\text{MCLR}}$	V_{SS}	0.2 V_{DD}	V	
D033		OSC1	V_{SS}	0.3 V_{DD}	V	HS 和 HSPLL 模式
D033A		OSC1	V_{SS}	0.2 V_{DD}	V	EC 模式
D034		T13CKI	V_{SS}	0.3	V	
	V_{IH}	输入高电压				
		I/O 端口：				
D040		带 TTL 缓冲器	0.25 V_{DD} + 0.8V	V_{DD}	V	
D041		带施密特触发缓冲器	0.8 V_{DD}	V_{DD}	V	
D042		$\overline{\text{MCLR}}$	0.8 V_{DD}	V_{DD}	V	
D043		OSC1	0.7 V_{DD}	V_{DD}	V	HS 和 HSPLL 模式
D043A		OSC1	0.8 V_{DD}	V_{DD}	V	EC 模式
D044		T13CKI	1.6	V_{DD}	V	
	I_{IL}	输入泄漏电流 (1)				
D060		I/O 端口	—	±1	μA	V_{SS} ≤ V_{PIN} ≤ V_{DD}， 引脚处于高阻状态
D061		$\overline{\text{MCLR}}$	—	±1	μA	V_{SS} ≤ V_{PIN} ≤ V_{DD}
D063		OSC1	—	±1	μA	V_{SS} ≤ V_{PIN} ≤ V_{DD}
	I_{PU}	弱上拉电流				
D070	I_{PURB}	PORTB 弱上拉电流	30	250	μA	V_{DD} = 3.3V， V_{PIN} = V_{SS}

注 1： 负电流定义为引脚的拉电流。

27.3 直流特性： PIC18F97J60 系列 （工业级） （续）

直流特性			标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ T_A ≤ +85°C			
参数编号	符号	特性	最小值	最大值	单位	条件
	V_{OL}	输出低电压				
D080		I/O 端口	—	0.6	V	I_{OL} = 8.5 mA， V_{DD} = 3.3V，-40°C 至 +85°C
D083		OSC2/CLKO （EC 和 ECPLL 模式）	—	0.6	V	I_{OL} = 1.6 mA， V_{DD} = 3.3V，-40°C 至 +85°C
	V_{OH}	输出高电压 [1]				
D090		I/O 端口	V_{DD} – 0.7	—	V	I_{OH} = -3.0 mA， V_{DD} = 3.3V，-40°C 至 +85°C
D092		OSC2/CLKO （内部 RC、 EC 和 ECPLL 模式）	V_{DD} – 0.7	—	V	I_{OH} = -1.3 mA， V_{DD} = 3.3V，-40°C 至 +85°C
		输出引脚上的容性负载规范				
D100	C_{OSC2}	OSC2 引脚	—	15	pF	当外部时钟用于驱动 OSC1 时处于 HS 模式下
D101	C_{IO}	所有 I/O 引脚和 OSC2 （在内部 RC、 EC 和 ECPLL 模式下）	—	50	pF	满足交流时序规范
D102	C_B	SCLx 和 SDAx	—	400	pF	I^2C™ 规范

注 1： 负电流定义为引脚的拉电流。

表 **27-1**： 存储器编程要求

直流特性			标准运行条件 （除非另外说明） 运行温度 工业级为 -40°C ≤ TA ≤ +85°C				
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
		程序闪存					
D130	EP	单元擦写次数	100	1K	—	E/W	-40°C 至 +85°C
D131	VPR	用于读取的 VDD	VMIN	—	3.6	V	VMIN = 最小工作电压
D132B	VPEW	自定时写的 VDD	VMIN	—	3.6	V	VMIN = 最小工作电压
D133A	TIW	自定时写周期时间	—	2.8	—	ms	
D134	TRETD	特性保存期	20	—	—	年	假设没有违反其他规范
D135	IDDP	编程期间的供电电流	—	10	—	mA	

† 除非另外说明，否则 “典型值” 栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考，未经测试。



表 27-2：　　比较器规范

运行条件：3.0V ≤ V_{DD} ≤ 3.6V， -40°C ≤ T_A ≤ +85°C （除非另外说明）							
参数编号	符号	特性	最小值	典型值	最大值	单位	注释
D300	V_{IOFF}	输入失调电压	—	±5.0	±10	mV	
D301	V_{ICM}	输入共模电压 *	0	—	AV_{DD} – 1.5	V	
D302	CMRR	共模抑制比 *	55	—	—	dB	
300	T_{RESP}	响应时间 (1)*	—	150	400	ns	
301	T_{MC2OV}	比较器模式变为输出有效的时间 *	—	—	10	µs	

* 这些参数为特性值，但未经测试。

注　1：响应时间是在比较器的一个输入端电压为 (AV_{DD} – 1.5)/2，而另一个输入端从 V_{SS} 跳变到 AV_{DD} 时测得的。

表 27-3：　　参考电压规范

运行条件：3.0V ≤ V_{DD} ≤ 3.6V， -40°C ≤ T_A ≤ +85°C （除非另外说明）							
参数编号	符号	特性	最小值	典型值	最大值	单位	注释
D310	V_{RES}	分辨率	V_{DD}/24	—	V_{DD}/32	LSb	
D311	V_{RAA}	绝对精度	—	—	1/2	LSb	
D312	V_{RUR}	单位电阻值 （R）	—	2k	—	Ω	
310	T_{SET}	稳定时间 (1)	—	—	10	µs	

注　1：稳定时间是在 CVRR = `1` 并且 CVR3:CVR0 从 `0000` 跳变到 `1111` 时测得的。

表 27-4：　　内部稳压器规范

运行条件：-40°C ≤ T_A ≤ +85°C （除非另外说明）							
参数编号	符号	特性	最小值	典型值	最大值	单位	注释
	V_{RGOUT}	稳压器输出电压	—	2.5	—	V	
	C_{EFC}	外部滤波电容值	1	10	—	µF	电容必须为低串联电阻

* 这些参数为特性值，但未经测试。尚未为这些规范分配参数编号。

27.4 交流（时序）特性

27.4.1 时序参数符号体系

时序参数符号采用以下格式之一进行创建：

1. TppS2ppS
2. TppS
3. TCC:ST（仅用于 I^2C 规范）
4. Ts（仅用于 I^2C 规范）

T			
F	频率	T	时间

小写字母（pp）及其含义：

pp			
cc	ECCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CSx}$	rw	$\overline{RD}$ 或 $\overline{WR}$
di	SDIx	sc	SCKx
do	SDOx	ss	$\overline{SSx}$
dt	数据输入	t0	T0CKI
io	I/O 端口	t1	T13CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

大写字母及其含义：

S			
F	下降	P	周期
H	高	R	上升
I	无效（高阻）	V	有效
L	低	Z	高阻
仅用于 I^2C			
AA	输出访问	High	高
BUF	总线空闲	Low	低

TCC:ST（仅用于 I^2C 规范）

CC			
HD	保持	SU	建立
ST			
DAT	数据输入保持	STO	停止条件
STA	启动条件		

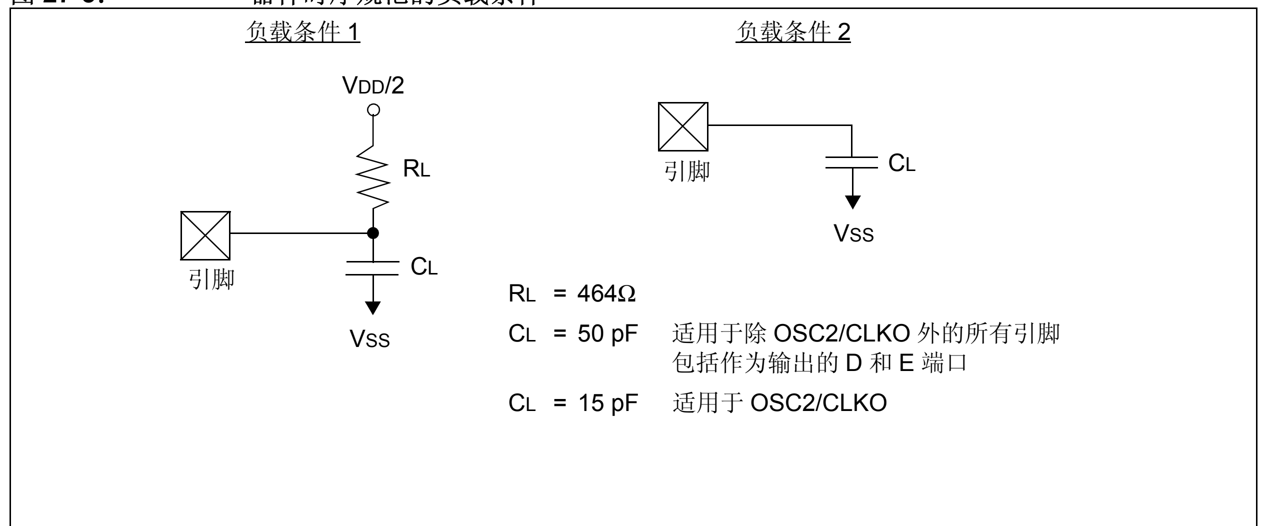
27.4.2 时序条件

表 27-5中指定的温度和电压适用于所有的时序规范（除非另外说明）。图 27-3 规定了时序规范的负载条件。

表 27-5： 温度和电压规范——交流

交流特性	标准运行条件 （除非另外说明） 运行温度　　工业级为 -40°C ≤ T_A ≤ +85°C 直流规范第 **27.1** 节和第 **27.3** 节描述了工作电压 V_{DD} 的范围。

图 27-3： 器件时序规范的负载条件



 超前信息

27.4.3 时序图和规范

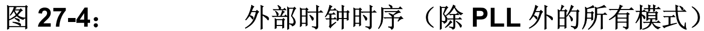


图 27-4： 外部时钟时序 （除 **PLL** 外的所有模式）

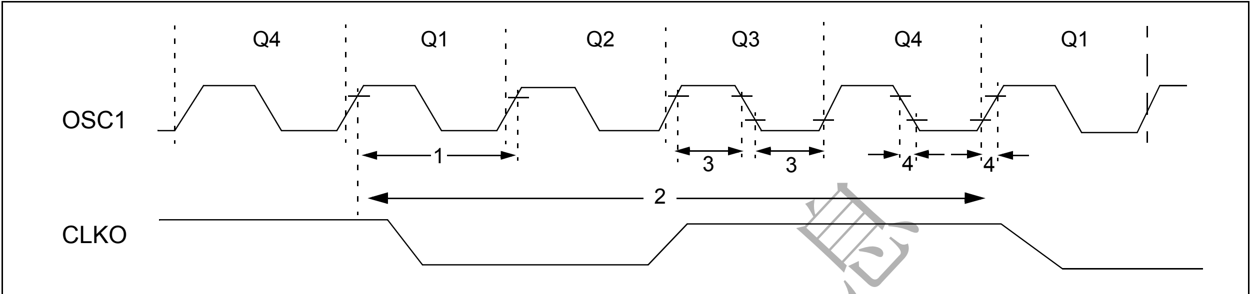


表 27-6： 外部时钟时序要求

参数编号	符号	特性	最小值	最大值	单位	条件
1A	FOSC	外部 CLKI 频率 [(1)]	DC	41.6667	MHz	EC 振荡器模式
		振荡器频率 [(1)]	6	25	MHz	HS 振荡器模式
1	TOSC	外部 CLKI 周期 [(1)]	24	—	ns	HS 振荡器模式
		振荡器周期 [(1)]	40	167	ns	HS 振荡器模式
2	TCY	指令周期时间 [(1)]	96	—	ns	TCY = 4/FOSC，工业级
3	TOSL，TOSH	外部时钟输入 （OSC1）的高电平或低电平时间	10	—	ns	EC 振荡器模式
4	TOSR，TOSF	外部时钟输入 （OSC1）的上升或下降时间	—	7.5	ns	EC 振荡器模式
5		时钟频率容差	—	±50	ppm	使能以太网模块

注 **1：** 对于除 PLL 外的所有配置，指令周期时间 （TCY）等于输入振荡器时基周期的四倍。所有规定值均为基于器件在标准运行条件下执行代码所对应的特定振荡器类型的特性数据。超出规定值可能导致振荡器运行不稳定和 / 或电流消耗超出预期值。所有器件在测试 “最小值”时，都在 OSC1/CLKI 引脚连接了外部时钟。当使用了外部时钟输入时，所有器件的 “最大”周期时间限制为 “DC”（无时钟）。

表 27-7： PLL 时钟时序规范 （VDD = 2.6V 至 3.6V）

参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
F10	FOSC	振荡器频率范围	8	—	25	MHz	HSPLL 模式
			8	—	37.5	MHz	仅 ECPLL 模式
F11	FSYS	片上 VCO 系统频率	20	—	62.5	MHz	
F12	t_{rc}	PLL 起振时间 （锁定时间）	—	—	2	ms	
F13	ΔCLK	CLKO 稳定性 （抗抖动）	-2	—	+2	%	

† 除非另外说明，否则 “典型值” 栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考，未经测试。

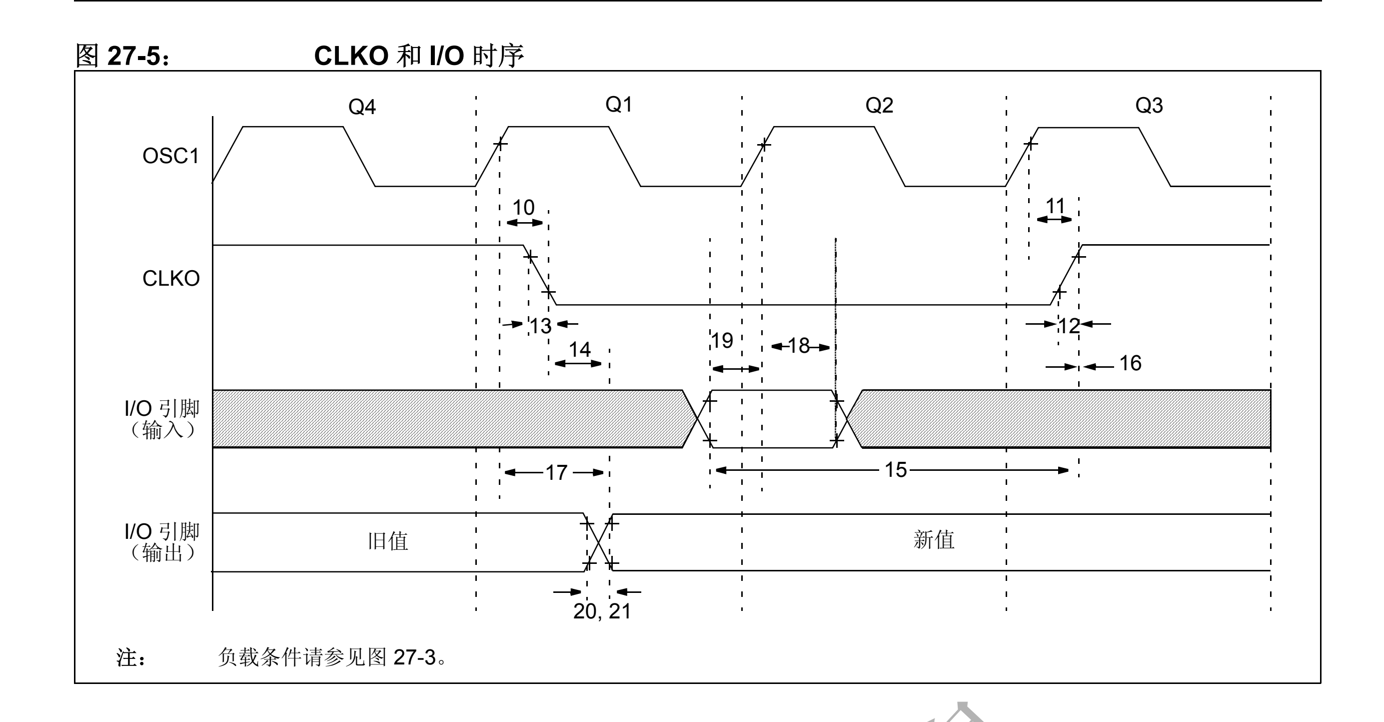
表 27-8： 交流特性： 内部 RC 精度 PIC18F97J60 系列 （工业级）

参数编号	特性	最小值	典型值	最大值	单位	条件
	在频率为 31 kHz(1) 时的 INTRC 精度	21.7	—	40.3	kHz	

注 1： INTRC 频率随 VDDCORE 的变化而改变。



图 27-5： CLKO 和 I/O 时序



注： 负载条件请参见图 27-3。

表 27-9： CLKO 和 I/O 时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
10	TosH2ckL	OSC1 ↑ 到 CLKO ↓ 的时间	—	75	200	ns	
11	TosH2ckH	OSC1 ↑ 到 CLKO ↑ 的时间	—	75	200	ns	
12	TckR	CLKO 上升时间	—	15	30	ns	
13	TckF	CLKO 下降时间	—	15	30	ns	
14	TckL2ioV	CLKO ↓ 到端口输出有效的时间	—	—	0.5 TCY + 20	ns	
15	TioV2ckH	在 CLKO ↑ 之前端口输入有效的时间	0.25 TCY + 25	—	—	ns	
16	TckH2ioI	在 CLKO ↑ 之后端口输入保持的时间	0	—	—	ns	
17	TosH2ioV	OSC1 ↑ （Q1 周期）到端口输出有效的时间	—	50	150	ns	
18	TosH2ioI	OSC1 ↑ （Q2 周期）到端口输入无效的时间 （I/O 输入保持时间）	100	—	—	ns	
19	TioV2osH	端口输入有效到 OSC1↑ 的时间（I/O 输入建立时间）	0	—	—	ns	
20	TioR	端口输出上升时间	—	—	6	ns	
21	TioF	端口输出下降时间	—	—	5	ns	
22†	TINP	INTx 引脚高电平或低电平时间	TCY	—	—	ns	
23†	TRBP	RB7:RB4 电平变化中断 INTx 高电平或低电平时间	TCY	—	—	ns	

† 这些参数是与任何内部时钟边沿无关的异步事件。

图 27-6： 读程序存储器时序图

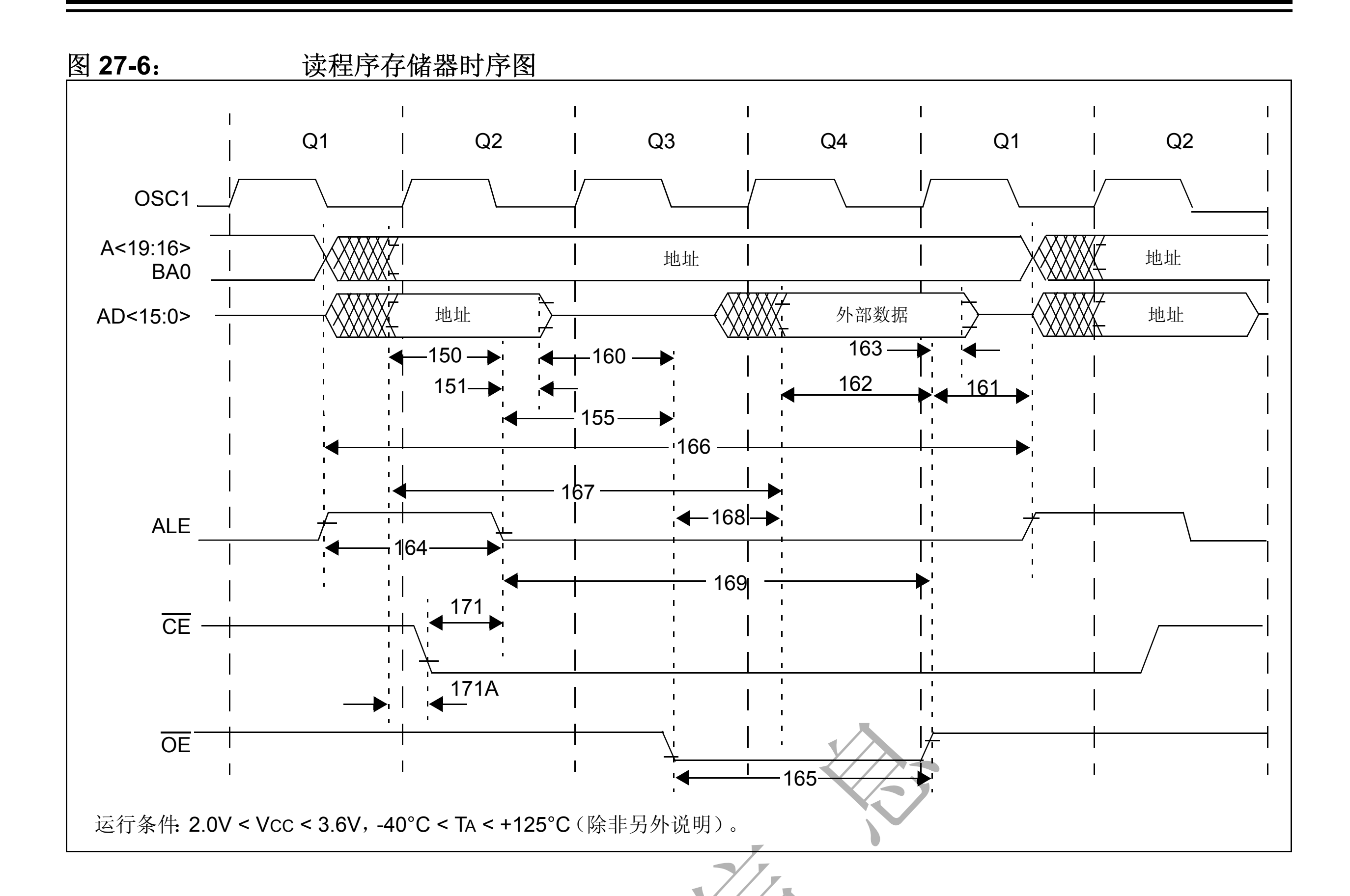


表 27-10： **CLKO 和 I/O 时序要求**

参数编号	符号	特性	最小值	典型值	最大值	单位
150	TadV2alL	地址输出有效到 ALE ↓ 的时间（地址建立时间）	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ 到地址输出无效的时间（地址保持时间）	5	—	—	ns
155	TalL2oeL	ALE ↓ 到 $\overline{OE}$ ↓ 的时间	10	0.125 TCY	—	ns
160	TadZ2oeL	AD 高阻态到 $\overline{OE}$ ↓ 的时间 （总线释放 OE）	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ 到驱动 AD 的时间	0.125 TCY – 5	—	—	ns
162	TadV2oeH	在 $\overline{OE}$ ↑ 之前的低位数据有效时间（数据建立时间）	20	—	—	ns
163	ToeH2adl	$\overline{OE}$ ↑ 到数据输入无效的时间（数据保持时间）	0	—	—	ns
164	TalH2alL	ALE 脉冲宽度	—	TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ 脉冲宽度	0.5 TCY – 5	0.5 TCY	—	ns
166	TalH2alH	ALE ↑ 到 ALE ↑ 的时间 （周期时间）	—	0.25 TCY	—	ns
167	Tacc	地址有效到数据有效的时间	0.75 TCY – 25	—	—	ns
168	Toe	$\overline{OE}$ ↓ 到数据有效的时间		—	0.5 TCY – 25	ns
169	TalL2oeH	ALE ↓ 到 $\overline{OE}$ ↑ 的时间	0.625 TCY – 10	—	0.625 TCY + 10	ns
171	TalH2csL	芯片使能有效到 ALE ↓ 的时间	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD 有效到芯片使能有效的时间	—	—	10	ns

图 27-7： 写程序存储器时序图

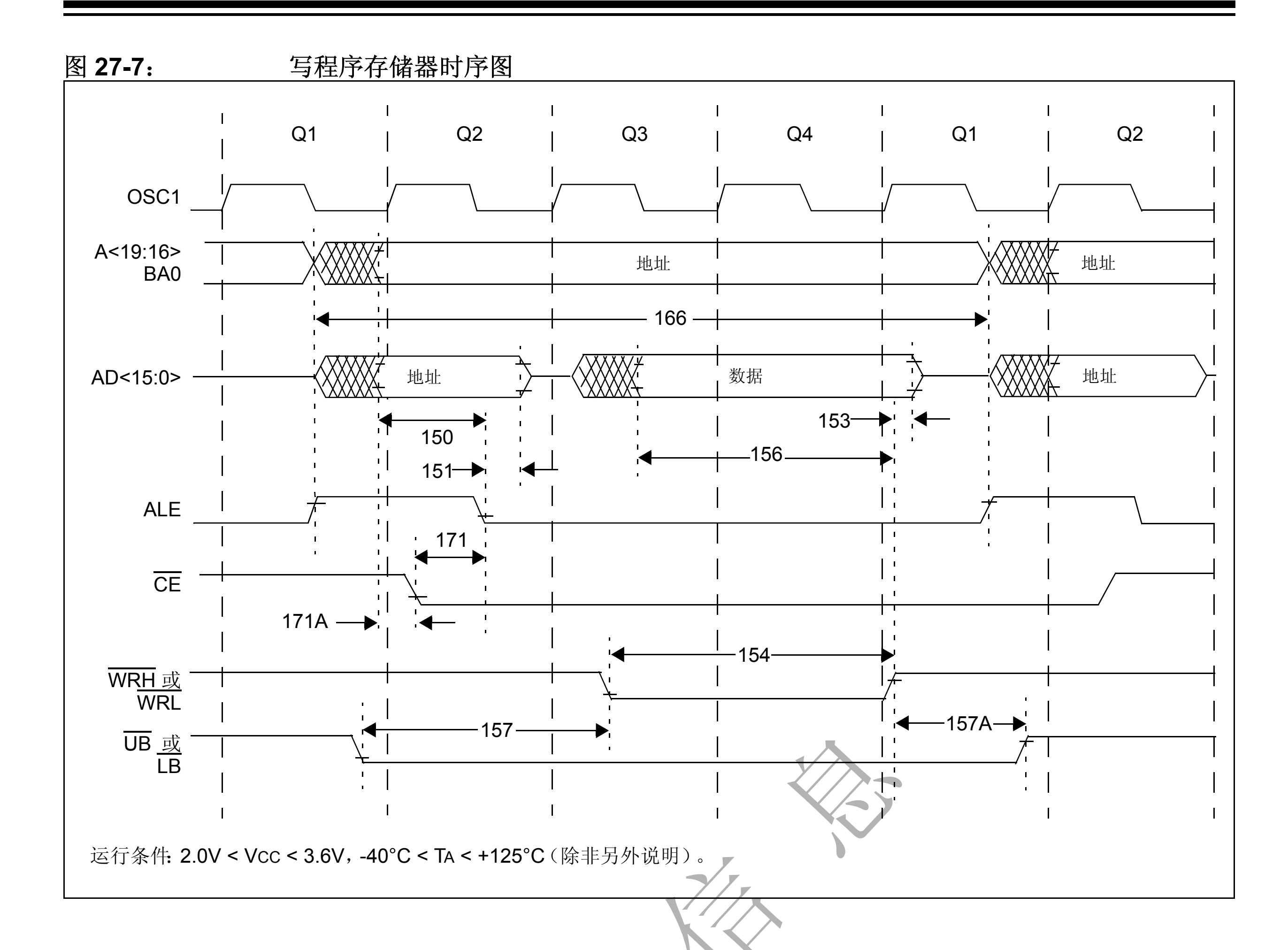
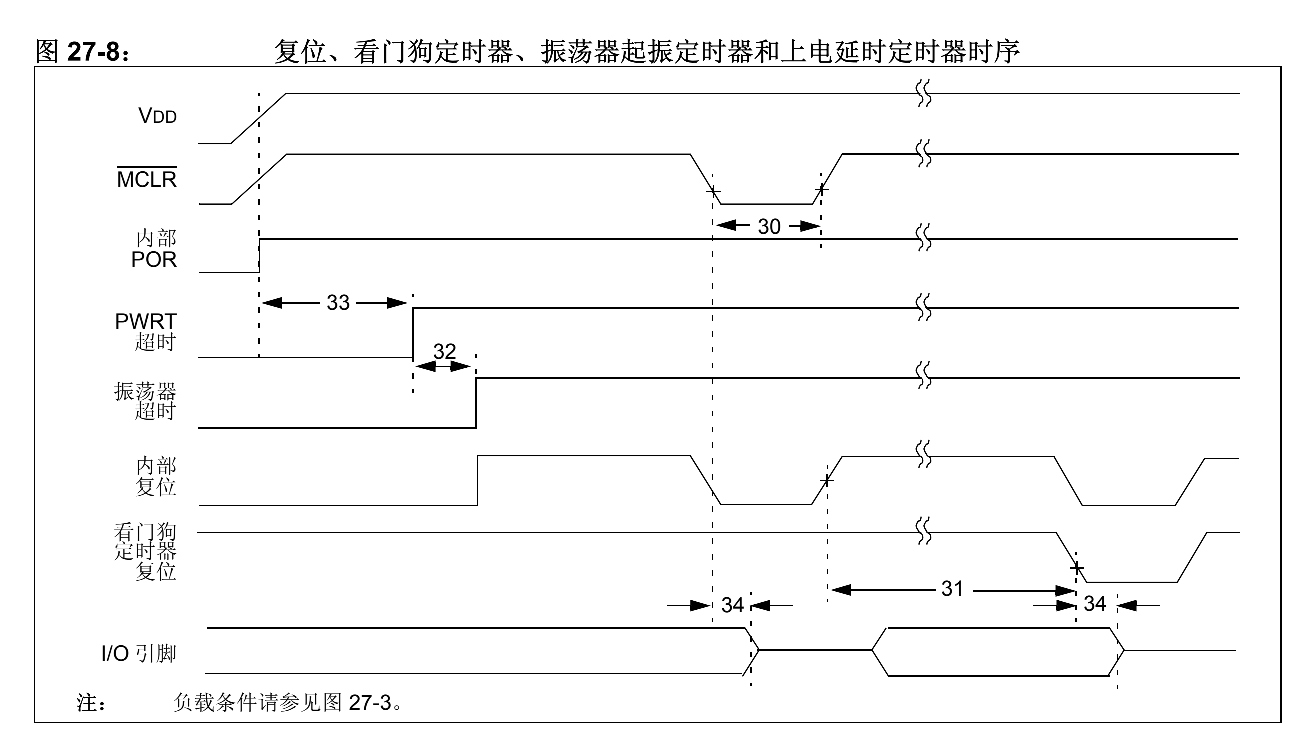


表 27-11： 写程序存储器时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位
150	TadV2alL	地址输出有效到 ALE ↓ 的时间 （地址建立时间）	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ 到地址输出无效的时间 （地址保持时间）	5	—	—	ns
153	TwrH2adl	$\overline{\text{WRn}}$ ↑ 到数据输出无效的时间 （数据保持时间）	5	—	—	ns
154	TwrL	$\overline{\text{WRn}}$ 脉冲宽度	0.5 TCY – 5	0.5 TCY	—	ns
156	TadV2wrH	在 $\overline{\text{WRn}}$ ↑ 之前的数据有效时间 （数据建立时间）	0.5 TCY – 10	—	—	ns
157	TbsV2wrL	在 $\overline{\text{WRn}}$ ↓ 之前的字节选择有效时间（字节选择建立时间）	0.25 TCY	—	—	ns
157A	TwrH2bsI	$\overline{\text{WRn}}$ ↑ 到字节选择无效的时间 （字节选择保持时间）	0.125 TCY – 5	—	—	ns
166	TalH2alH	ALE ↑ 到 ALE ↑ 的时间 （周期时间）	—	0.25 TCY	—	ns
171	TalH2csL	芯片使能有效到 ALE ↓ 的时间	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD 有效到芯片使能有效的时间	—	—	10	ns

图 27-8： 复位、看门狗定时器、振荡器起振定时器和上电延时定时器时序



注： 负载条件请参见图 27-3。

图 27-9： 欠压复位时序

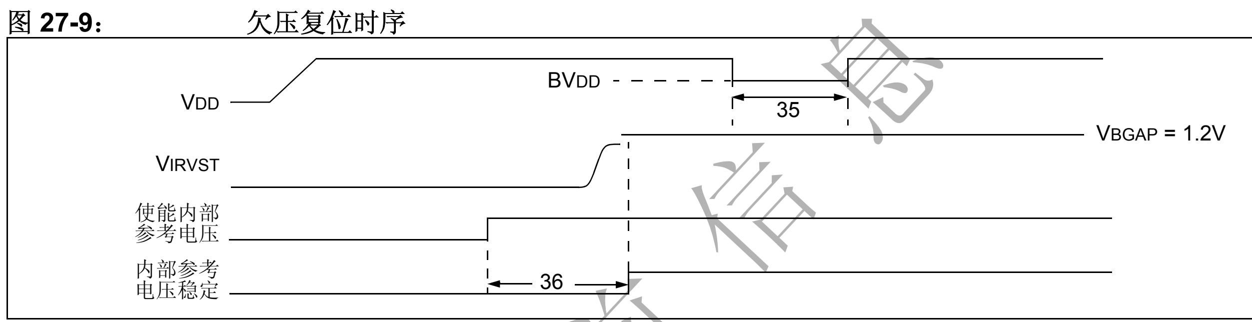


表 27-12： 复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
30	TMCL	$\overline{\text{MCLR}}$ 脉冲宽度 （低电平）	2	—	—	µs	
31	TWDT	看门狗定时器超时周期（无后分频器）	2.8	4.1	5.4	ms	
32	TOST	振荡器起振定时器周期	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 周期
33	TPWRT	上电延时定时器周期	46.2	66	85.8	ms	
34	TIOZ	自 $\overline{\text{MCLR}}$ 低电平或看门狗定时器复位起 I/O 处于高阻状态的时间	—	2	—	µs	
38	TCSD	CPU 启动时间	—	TBD	—	µs	

图注： TBD = 待定

图 27-10： TIMER0 和 TIMER1 外部时钟时序

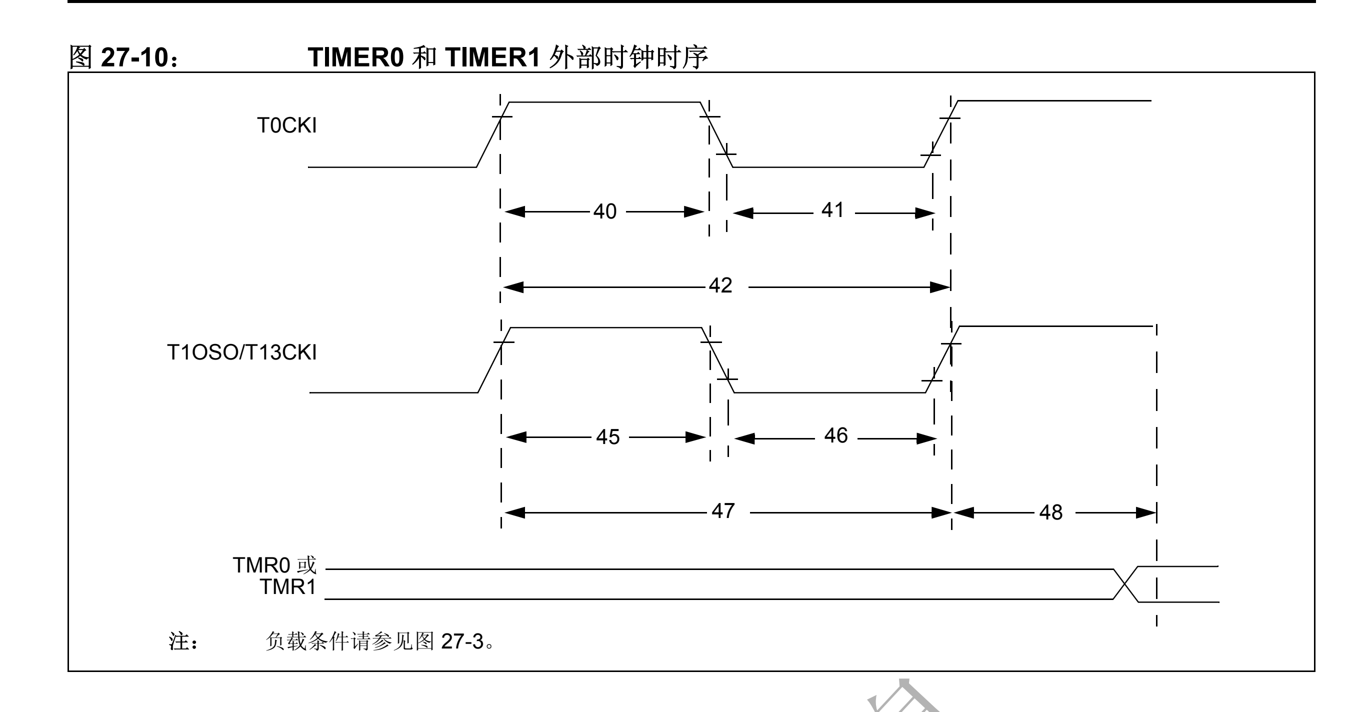
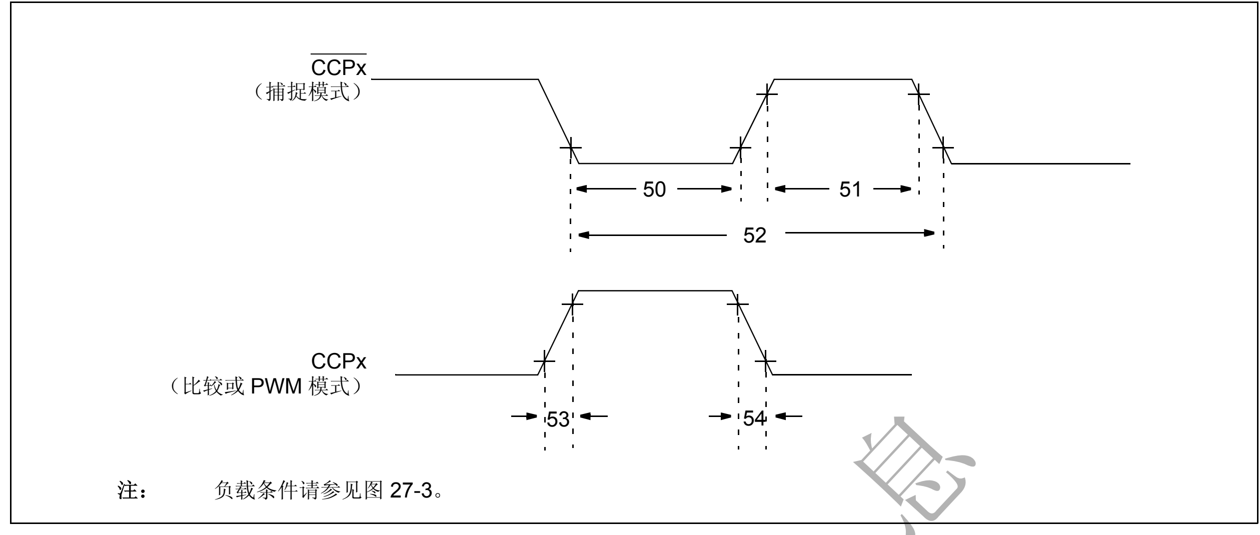


表 27-13： TIMER0 和 TIMER1 外部时钟要求

参数编号	符号	特性		最小值	最大值	单位	条件
40	TT0H	T0CKI 高电平脉冲宽度	无预分频器	0.5 TCY + 20	—	ns	
			有预分频器	10	—	ns	
41	TT0L	T0CKI 低电平脉冲宽度	无预分频器	0.5 TCY + 20	—	ns	
			有预分频器	10	—	ns	
42	TT0P	T0CKI 周期	无预分频器	TCY + 10	—	ns	
			有预分频器	取较大值：20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1, 2, 4,..., 256）
45	TT1H	T13CKI 高电平时间	同步，无预分频器	0.5 TCY + 20	—	ns	
			同步，有预分频器	10	—	ns	
			异步	30	—	ns	
46	TT1L	T13CKI 低电平时间	同步，无预分频器	0.5 TCY + 5	—	ns	
			同步，有预分频器	10	—	ns	
			异步	30	—	ns	
47	TT1P	T13CKI 输入周期	同步	取较大值：20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1, 2, 4, 8）
			异步	60	—	ns	
	FT1	T13CKI 振荡器输入频率范围		DC	50	kHz	
48	TCKE2TMRI	从外部 T13CKI 时钟边沿到定时器递增的延时		2 TOSC	7 TOSC	—	



图 27-11：　　捕捉 / 比较 /PWM 时序 （包括 ECCP 模块）



注：　负载条件请参见图 27-3。

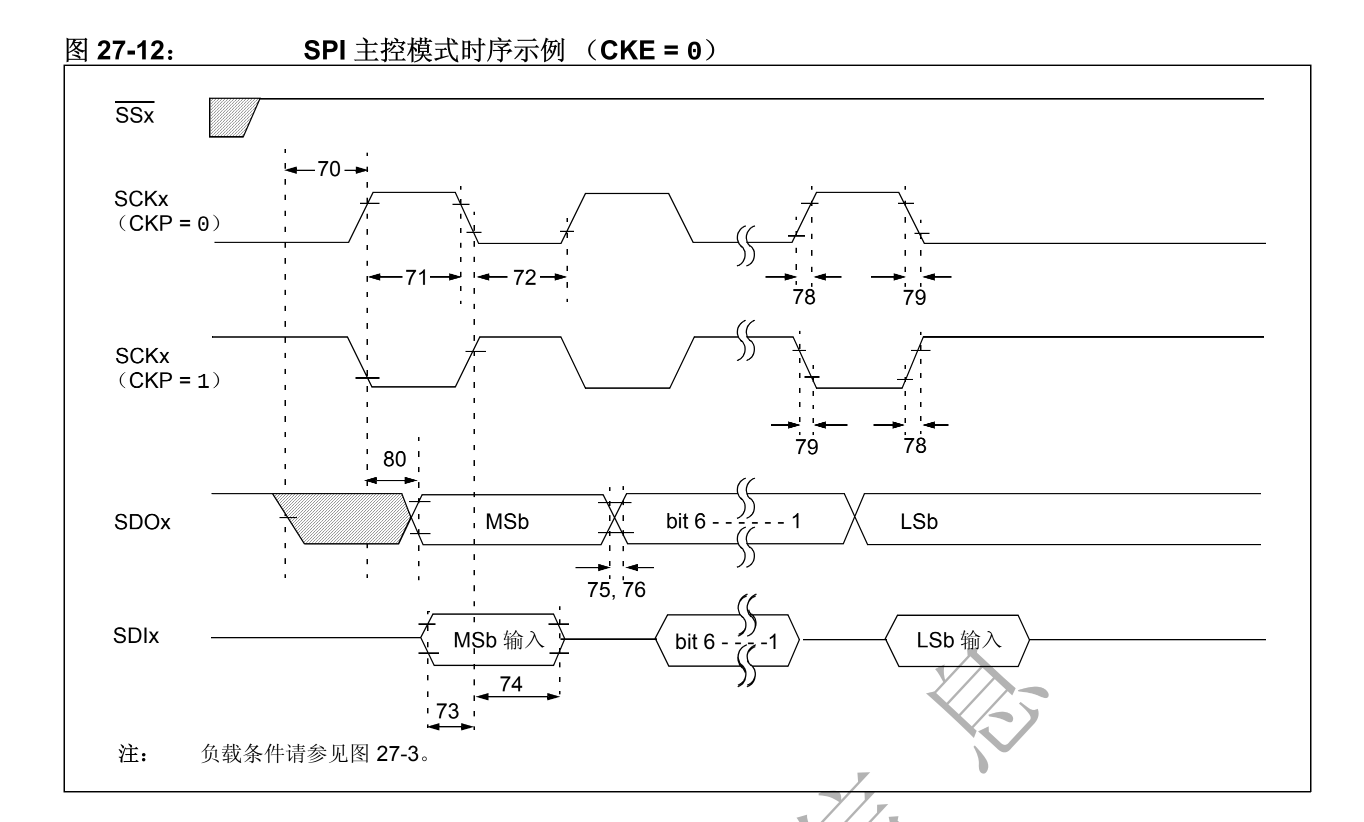
表 27-14：　　捕捉 / 比较 /PWM 要求 （包括 ECCP 模块）

参数编号	符号	特性		最小值	最大值	单位	条件
50	TccL	CCPx 输入低电平时间	无预分频器	0.5 TCY + 20	—	ns	
			有预分频器	10	—	ns	
51	TccH	CCPx 输入高电平时间	无预分频器	0.5 TCY + 20	—	ns	
			有预分频器	10	—	ns	
52	TccP	CCPx 输入周期		(3 TCY + 40) / N	—	ns	N = 预分频值（1、 4 或 16）
53	TccR	CCPx 输出下降时间		—	25	ns	
54	TccF	CCPx 输出下降时间		—	25	ns	

表 27-15：　　并行从动端口要求

参数编号	符号	特性	最小值	最大值	单位	条件
62	TdtV2wrH	在 $\overline{WR}$ ↑或 $\overline{CS}$ ↑之前的数据输入有效时间（建立时间）	20	—	ns	
63	TwrH2dtI	$\overline{WR}$ ↑或 $\overline{CS}$ ↑到数据输入无效的时间（保持时间）	20	—	ns	
64	TrdL2dtV	$\overline{RD}$ ↓和 $\overline{CS}$ ↓到数据输出有效的时间	—	80	ns	
65	TrdH2dtI	$\overline{RD}$ ↑或 $\overline{CS}$ ↓到数据输出无效的时间	10	30	ns	
66	TibfINH	禁止 IBF 标志位被 $\overline{WR}$ ↑或 $\overline{CS}$ ↑清零	—	3 TCY		

图 27-12： SPI 主控模式时序示例 （CKE = 0）



注： 负载条件请参见图 27-3。

表 27-16： SPI 模式要求示例 （主控模式， CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH， TssL2scL	SSx ↓ 到 SCKx ↓ 或 SCKx ↑ 输入的时间		TCY	—	ns	
71	TscH	SCKx 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH， TdiV2scL	SDIx 数据输入到 SCKx 边沿的建立时间		100	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY + 40	—	ns	（注 2）
74	TscH2diL， TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	ns	
75	TdoR	SDOx 数据输出上升时间		—	25	ns	
76	TdoF	SDOx 数据输出下降时间		—	25	ns	
78	TscR	SCKx 输出上升时间 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV， TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	

注 **1**： 要求使用参数 #73A。
2： 仅当使用参数 #71A 和 #72A 时。

图 27-13： SPI 主控模式时序示例 （CKE = 1）

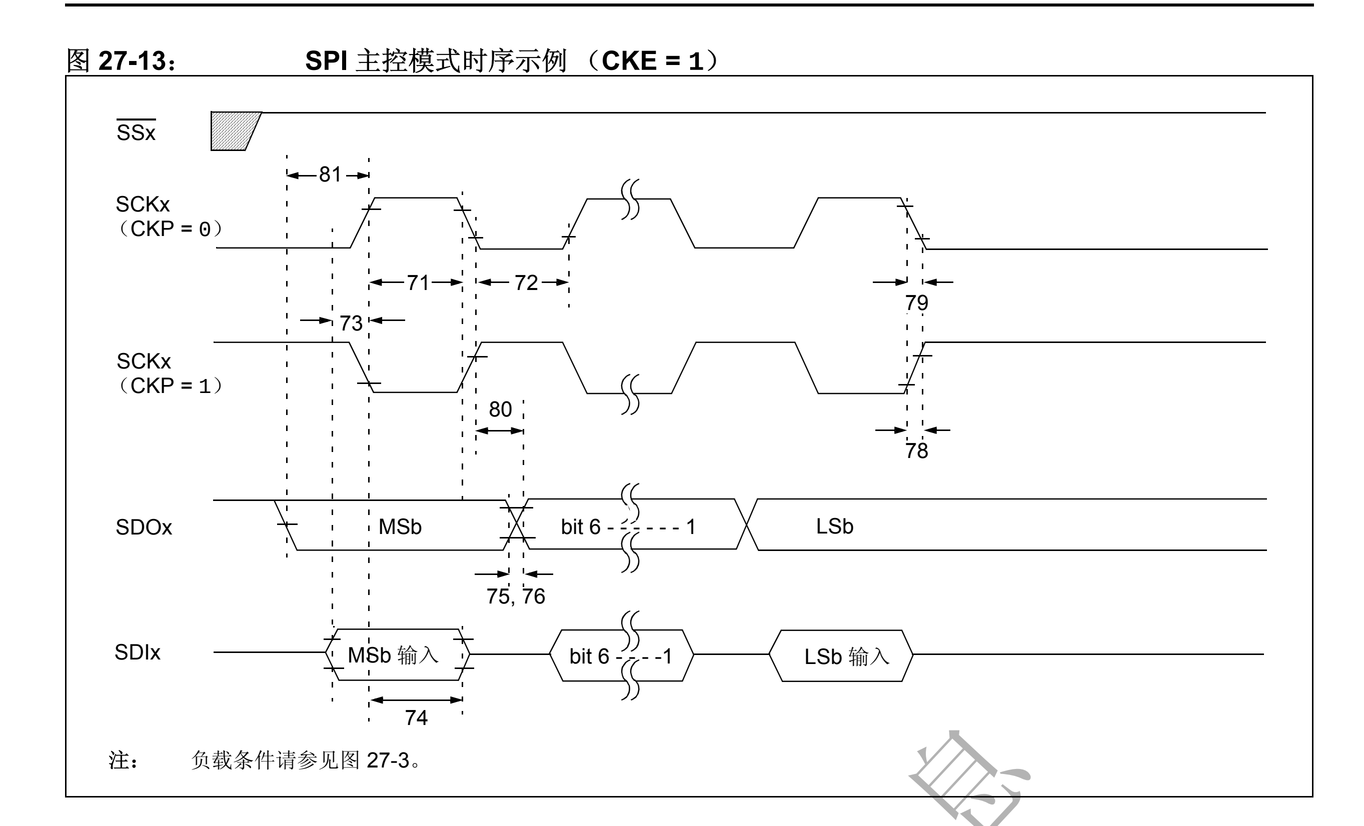


表 27-17： SPI 模式要求示例 （主控模式， CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
71	TscH	SCKx 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDIx 数据输入到 SCKx 边沿的建立时间		100	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY + 40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	ns	
75	TdoR	SDOx 数据输出上升时间		—	25	ns	
76	TdoF	SDOx 数据输出下降时间		—	25	ns	
78	TscR	SCKx 输出上升时间 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
81	TdoV2scH，TdoV2scL	SDOx 数据输出建立到出现 SCKx 边沿的时间		TCY	—	ns	

注 1： 要求使用参数 #73A。
2： 仅当使用参数 #71A 和 #72A 时。

图 27-14： SPI 从动模式时序示例 （CKE = 0）

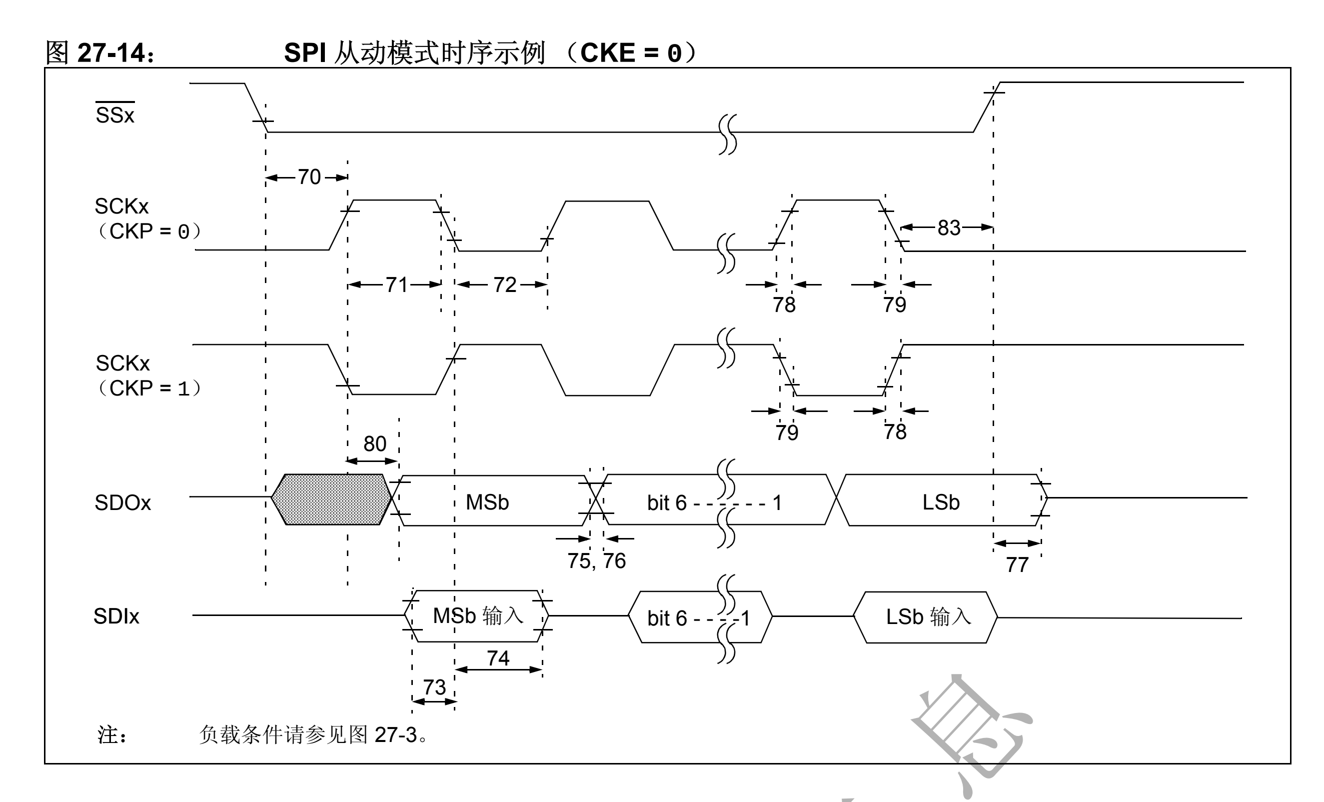


表 27-18： SPI 模式要求示例 （从动模式时序， CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SSx ↓ 到 SCKx ↓ 或 SCKx ↑ 输入的时间		TCY	—	ns	
71	TscH	SCKx 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDIx 数据输入到 SCKx 边沿的建立时间		100	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY + 40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	ns	
75	TdoR	SDOx 数据输出上升时间		—	25	ns	
76	TdoF	SDOx 数据输出下降时间		—	25	ns	
77	TssH2doZ	SSx ↑ 到 SDOx 输出高阻态的时间		10	50	ns	
78	TscR	SCKx 输出上升时间 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
83	TscH2ssH，TscL2ssH	在 SCKx 边沿之后出现 SSx ↑ 的时间		1.5 TCY + 40	—	ns	

注 1： 要求使用参数 #73A。
2： 仅当使用参数 #71A 和 #72A 时。

图 27-15：　SPI 从动模式时序示例 （CKE = 1）

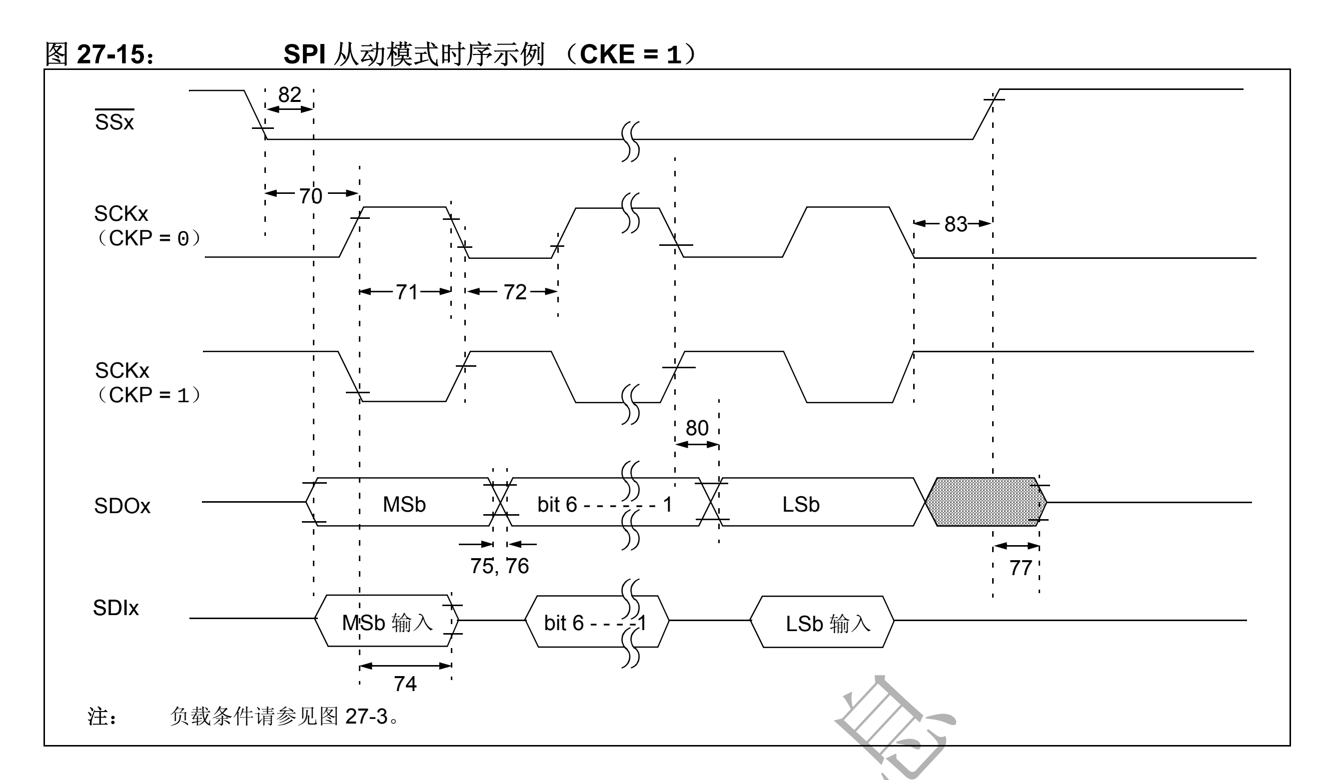
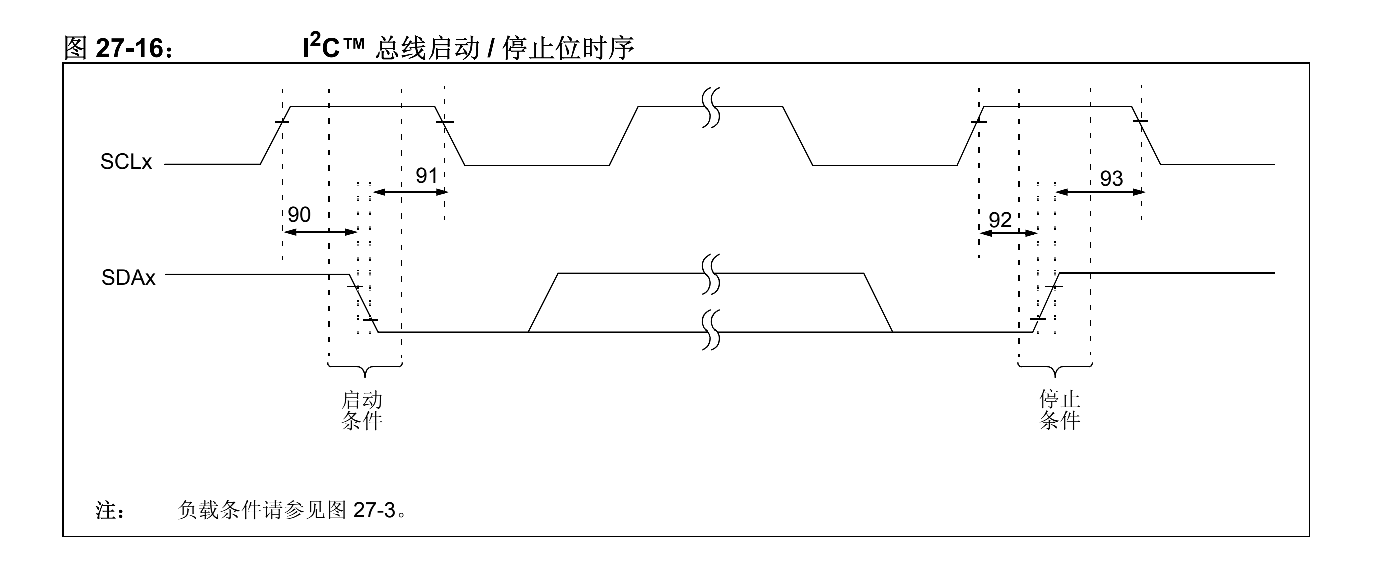


表 27-19：　SPI 从动模式要求示例 （CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	$\overline{SSx}$ ↓ 到 SCKx ↓ 或 SCKx ↑ 输入的时间		TCY	—	ns	
71	TscH	SCKx 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY + 40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	ns	
75	TdoR	SDOx 数据输出上升时间		—	25	ns	
76	TdoF	SDOx 数据输出下降时间		—	25	ns	
77	TssH2doZ	$\overline{SSx}$ ↑ 到 SDOx 输出高阻态的时间		10	50	ns	
78	TscR	SCKx 输出上升时间 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
82	TssL2doV	在 $\overline{SSx}$ ↓ 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
83	TscH2ssH，TscL2ssH	在 SCKx 边沿之后出现 $\overline{SSx}$ ↑ 的时间		1.5 TCY + 40	—	ns	

注　**1：**　要求使用参数 #73A。
　　2：　仅当使用参数 #71A 和 #72A 时。

图 27-16： I²C™ 总线启动 / 停止位时序

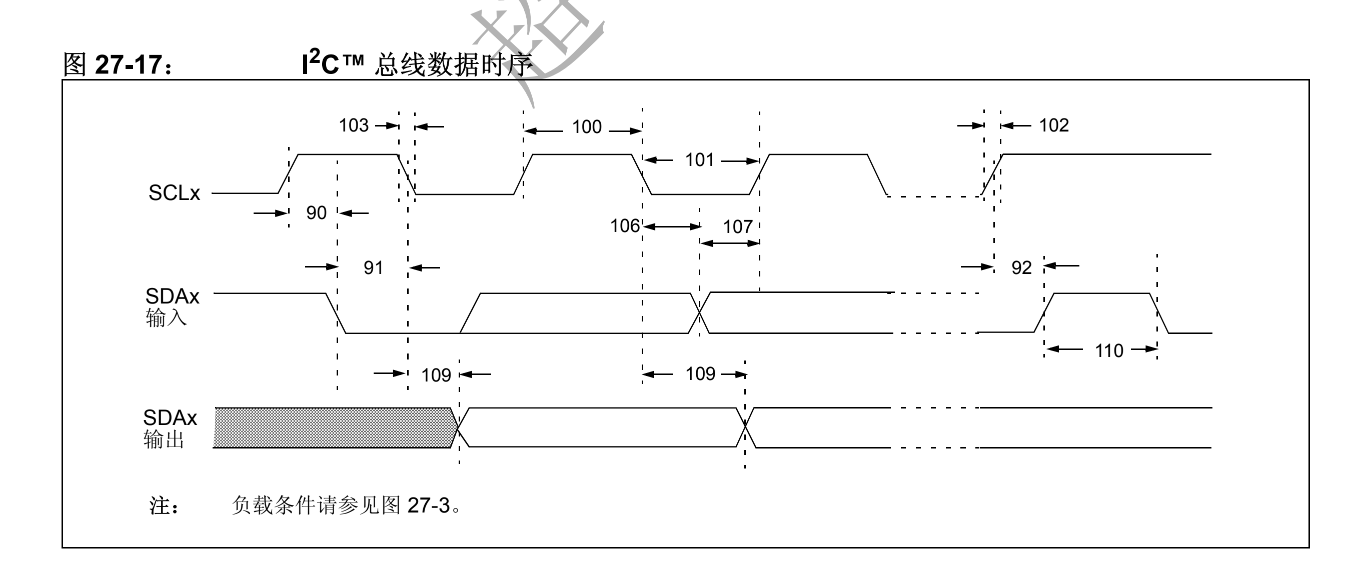


注： 负载条件请参见图 27-3。

表 27-20： I²C™ 总线启动 / 停止位要求 （从动模式）

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件	100 kHz 模式	4700	—	ns	仅与重复启动条件相关
		建立时间	400 kHz 模式	600	—		
91	THD:STA	启动条件	100 kHz 模式	4000	—	ns	这个周期后产生第一个时钟脉冲
		保持时间	400 kHz 模式	600	—		
92	TSU:STO	停止条件	100 kHz 模式	4700	—	ns	
		建立时间	400 kHz 模式	600	—		
93	THD:STO	停止条件	100 kHz 模式	4000	—	ns	
		保持时间	400 kHz 模式	600	—		

图 27-17： I²C™ 总线数据时序



注： 负载条件请参见图 27-3。

表 27-21： I^2C™ 总线数据要求 （从动模式）

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	μs	PIC18F97J60 系列的工作频率不得低于 1.5 MHz
			400 kHz 模式	0.6	—	μs	PIC18F97J60 系列的工作频率不得低于 10 MHz
			MSSP 模块	1.5 TCY	—		
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	μs	PIC18F97J60 系列的工作频率不得低于 1.5 MHz
			400 kHz 模式	1.3	—	μs	PIC18F97J60 系列的工作频率不得低于 10 MHz
			MSSP 模块	1.5 TCY	—		
102	TR	SDAx 和 SCLx 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值的范围在 10 至 400 pF 之间
103	TF	SDAx 和 SCLx 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值的范围在 10 至 400 pF 之间
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	μs	仅与重复启动条件相关
			400 kHz 模式	0.6	—	μs	
91	THD:STA	启动条件保持时间	100 kHz 模式	4.0	—	μs	这个周期后产生第一个时钟脉冲
			400 kHz 模式	0.6	—	μs	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	μs	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	（注 **2**）
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	4.7	—	μs	
			400 kHz 模式	0.6	—	μs	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	（注 **1**）
			400 kHz 模式	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	μs	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	μs	
D102	CB	总线容性负载		—	400	pF	

注 **1**： 为避免产生意外的启动或停止条件，作为发送器的器件必须提供这个内部最小延时以避开 SCLx 下降沿的未定义区域（最小值 300 ns）。

2： 快速模式的 I^2C™ 总线器件也可在标准模式的 I^2C 总线系统中使用，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果快速模式器件没有延长 SCLx 信号的低电平时间，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平时间，其下一个数据位必须输出到 SDAx 线。 SCLx 线被释放前，根据标准模式 I^2C 总线规范， TR max. + TSU:DAT = 1000 + 250 = 1250 ns。

图 **27-18：** 主控 **SSP I²C™** 总线启动 **/** 停止位时序波形

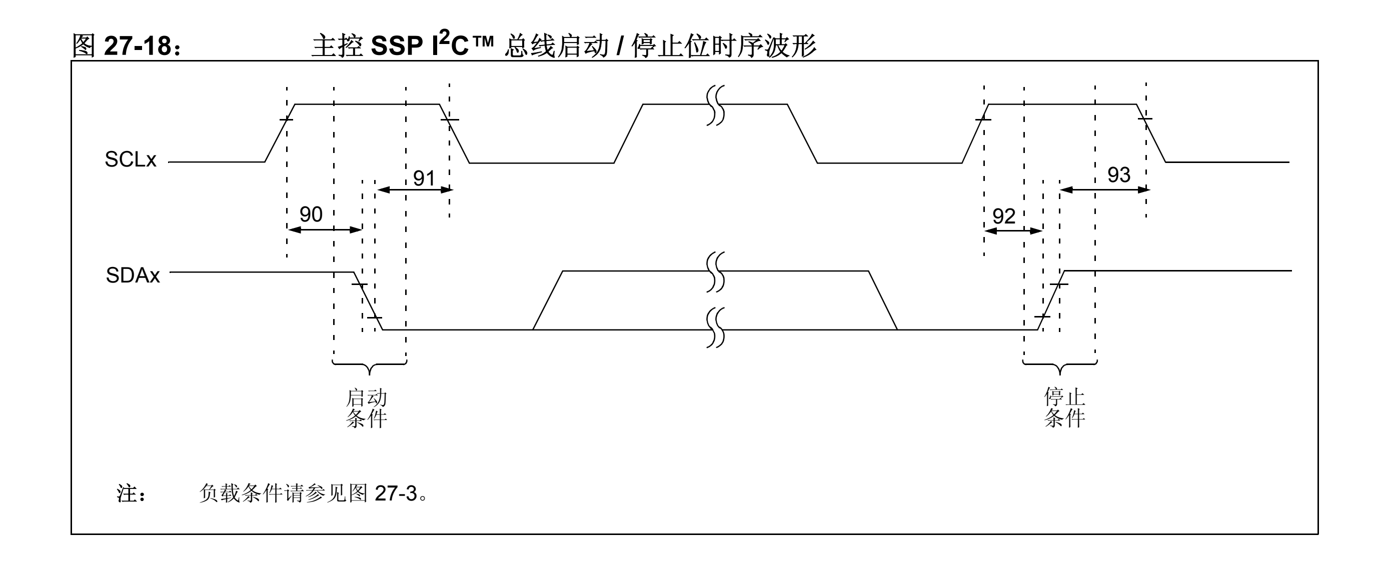


表 **27-22：** 主控 **SSP I²C™** 总线启动 **/** 停止位要求

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	仅与重复启动条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		

注 **1：** 对于所有 I²C™ 引脚，最大引脚电容均为 10 pF。

图 **27-19：** 主控 **SSP I²C™** 总线数据时序

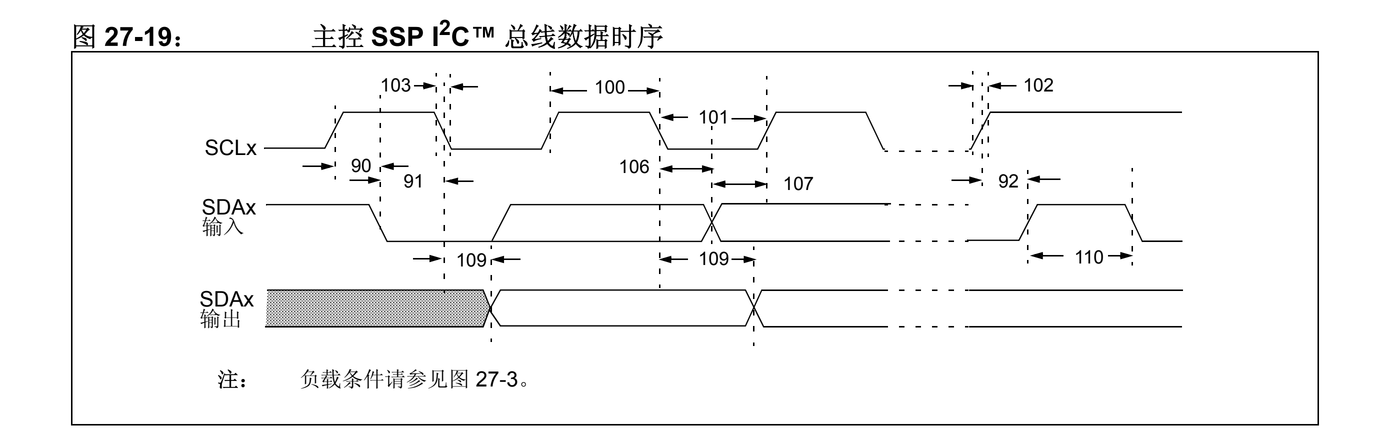


表 27-23： 主控 SSP I²C™ 总线数据要求

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	时钟低电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDAx 和 SCLx 上升时间	100 kHz 模式	—	1000	ns	CB 值的范围在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	300	ns	
103	TF	SDAx 和 SCLx 下降时间	100 kHz 模式	—	300	ns	CB 值的范围在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	100	ns	
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	仅与重复启动条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
			1 MHz 模式 (1)	TBD	—	ns	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	（注 2）
			400 kHz 模式	100	—	ns	
			1 MHz 模式 (1)	TBD	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	
			400 kHz 模式	—	1000	ns	
			1 MHz 模式 (1)	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	ms	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	ms	
			1 MHz 模式 (1)	TBD	—	ms	
D102	CB	总线容性负载		—	400	pF	

图注： TBD = 待定

注 1： 对于所有 I²C™ 引脚，最大引脚电容均为 10 pF。

2： 快速模式的 I²C 总线器件也可在标准模式的 I²C 总线系统中使用，但必须满足参数 #107 ≥ 250 ns 的要求。如果快速模式器件没有延长 SCLx 信号的低电平时间，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平时间，它必须将下一个数据位输出到 SDAx 线。 SCLx 线被释放前，在 100 kHz 模式下，参数 #102 + 参数 #107 = 1000 + 250 = 1250 ns。

图 **27-20：** **EUSART** 同步发送 （主控 / 从动）时序

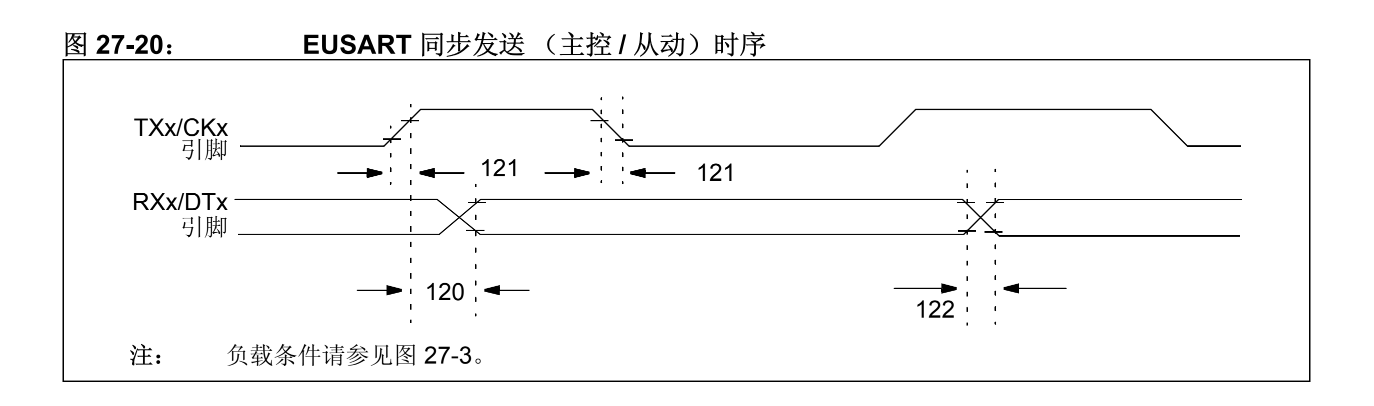


表 **27-24：** **EUSART** 同步发送要求

参数编号	符号	特性	最小值	最大值	单位	条件
120	TCKH2DTV	同步 XMIT （主控和从动） 时钟高电平到数据输出有效的时间	—	40	ns	
121	TCKRF	时钟输出上升时间和下降时间 （主控模式）	—	20	ns	
122	TDTRF	数据输出上升时间和下降时间	—	20	ns	

图 **27-21：** **EUSART** 同步接收 （主控 / 从动）时序

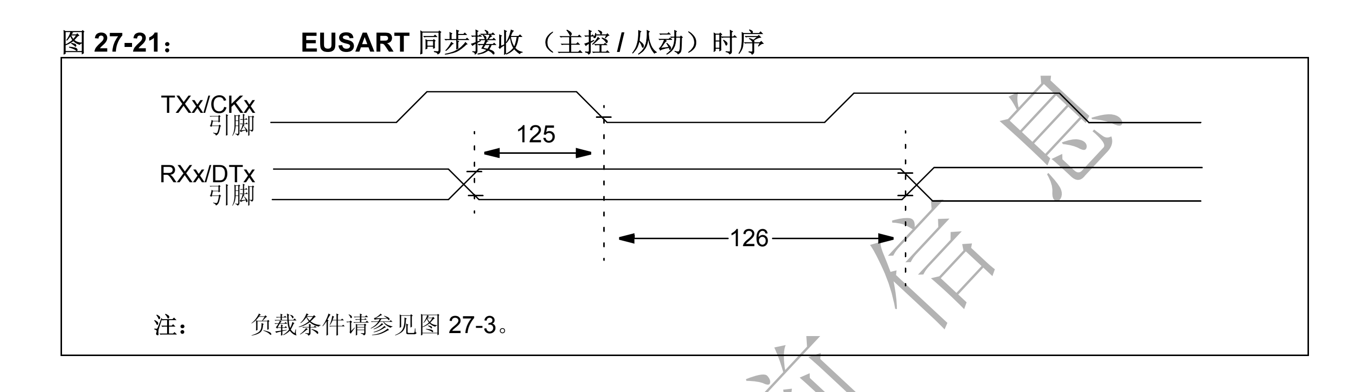


表 **27-25：** **EUSART** 同步接收要求

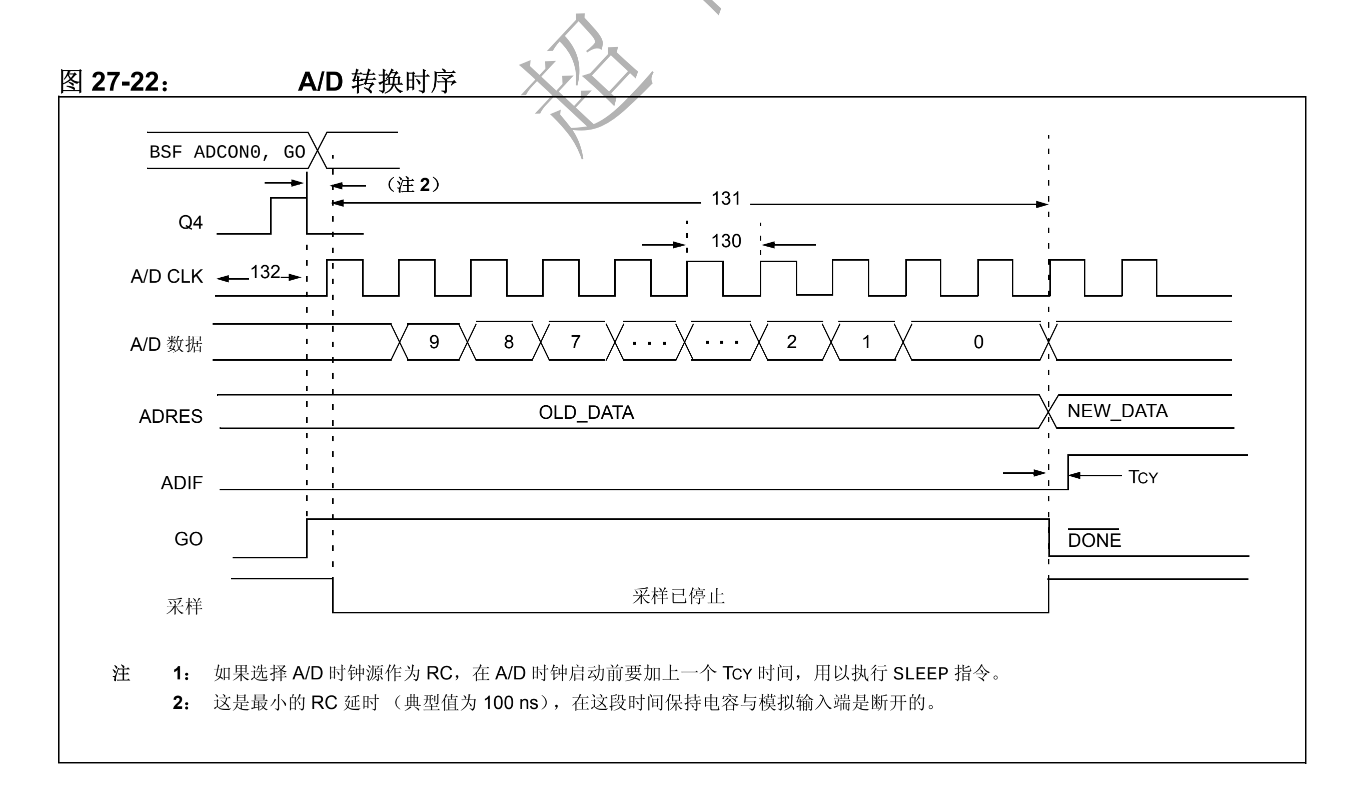
参数编号	符号	特性	最小值	最大值	单位	条件
125	TDTV2CKL	同步接收 （主控和从动） 在 CKx ↓ 之前数据的保持时间 （DTx 保持时间）	10	—	ns	
126	TCKL2DTL	在 CKx ↓ 之后数据的保持时间 （DTx 保持时间）	15	—	ns	

表 27-26： A/D 转换器特性：PIC18F97J60 系列 （工业级）

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
A01	NR	分辨率	—	—	10	位	ΔVREF ≥ 2.0V
A03	EIL	积分线性误差	—	—	<±1	LSb	ΔVREF ≥ 2.0V
A04	EDL	微分线性误差	—	—	<±1	LSb	ΔVREF ≥ 2.0V
A06	EOFF	失调误差	—	—	<±3	LSb	ΔVREF ≥ 2.0V
A07	EGN	增益误差	—	—	<±3	LSb	ΔVREF ≥ 2.0V
A10	—	单调性	保证 (1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	参考电压范围（VREFH – VREFL）	1.8 3	— —	— —	V V	VDD < 3.0V VDD ≥ 3.0V
	VREFSUM	参考电压总和（VREFH + VREFL）	—	—	AVDD+0.5	V	
A21	VREFH	参考电压高电平	VREFL	—	AVDD	V	
A22	VREFL	参考电压低电平	AVSS	—	VREFH	V	
A25	VAIN	模拟输入电压	VREFL	—	VREFH	V	
A30	ZAIN	模拟电源阻抗的推荐值	—	—	2.5	kΩ	
A50	IREF	VREF 输入电流 (2)	— —	— —	5 1000	μA μA	在采集 VAIN 期间。 在 A/D 转换周期期间。

注 1： A/D 转换结果不会因输入电压的增加而减小，并且不会丢失代码。
2： VREFH 电流来自作为 VREFH 源的 RA3/AN3/VREF+ 引脚或 AVDD。
VREFL 电流来自作为 VREFL 源的 RA2/AN2/VREF- 引脚或 AVSS。

图 27-22： A/D 转换时序



注 1： 如果选择 A/D 时钟源作为 RC，在 A/D 时钟启动前要加上一个 TCY 时间，用以执行 SLEEP 指令。
2： 这是最小的 RC 延时 （典型值为 100 ns），在这段时间保持电容与模拟输入端是断开的。

表 27-27： A/D 转换要求

参数编号	符号	特性	最小值	最大值	单位	条件
130	TAD	A/D 时钟周期	0.7	25.0[1]	μs	基于 TOSC， VREF ≥ 2.0V
			TBD	1	μs	A/D RC 模式
131	TCNV	转换时间 （不包括采集时间）（注 2）	11	12	TAD	
132	TACQ	采集时间 （注 3）	1.4 TBD	— —	μs μs	-40°C 至 +85°C 0°C 至 +85°C
135	TSWC	转换 → 采样的切换时间	—	（注 4）		
TBD	TDIS	电容放电时间	0.2	—	μs	

图注： TBD = 待定

注 1： A/D 时钟周期取决于器件频率和 TAD 时钟分频比。

2： 可在后续 TCY 周期内读 ADRES 寄存器。

3： 转换完成后当电压满量程变化时 （VDD 至 VSS 或 VSS 至 VDD），保持电容采样一个 “新”输入电压所需的时间。在输入通道上的电源阻抗 （RS）为 50Ω。

4： 在器件时钟的下一个周期。

27.5 以太网规范和要求

表 27-28： 以太网收发器外部磁场的要求

参数	最小值	正常值	最大值	单位	条件
RXx 匝数比	—	1:1	—	—	
TXx 匝数比	—	1:1	—	—	变压器中心抽头 = 3.3V
插入损耗	0.0	0.6	1.1	dB	
初级电感	350	—	—	μH	8 mA 偏置电流
变压器隔离	—	1.5	—	kVrms	
差分共模抑制	40	—	—	dB	0.1 至 10 MHz
回波损耗	-16	—	—	dB	

注：

超前信息

28.0 直流和交流特性图表

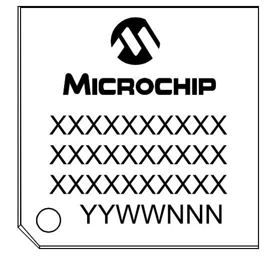
当前没有可用图表。

注：

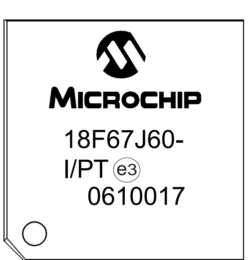
29.0 封装信息

29.1 封装标识信息

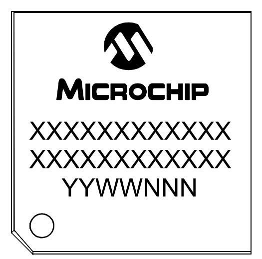
64 引脚 TQFP



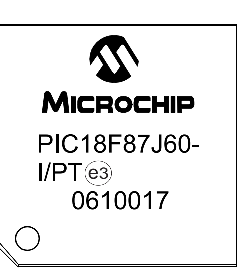
示例



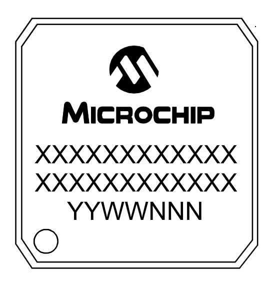
80 引脚 TQFP



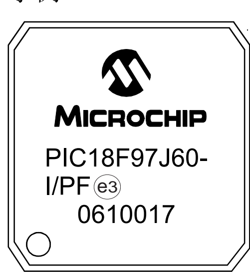
示例



100 引脚 TQFP



示例



图注：		
	XX...X	客户指定信息
	Y	年代码 （公历年份的最后一位）
	YY	年代码 （公历年份的最后两位）
	WW	星期代码 （1 月 1 日的星期代码为 “01”）
	NNN	字母数字的追踪代码
	(e3)	雾锡 （Sn）的无铅 JEDEC 标志
	*	本封装为无铅封装。在封装的外部包装上可以找到无铅 JEDEC 标志（(e3)）。

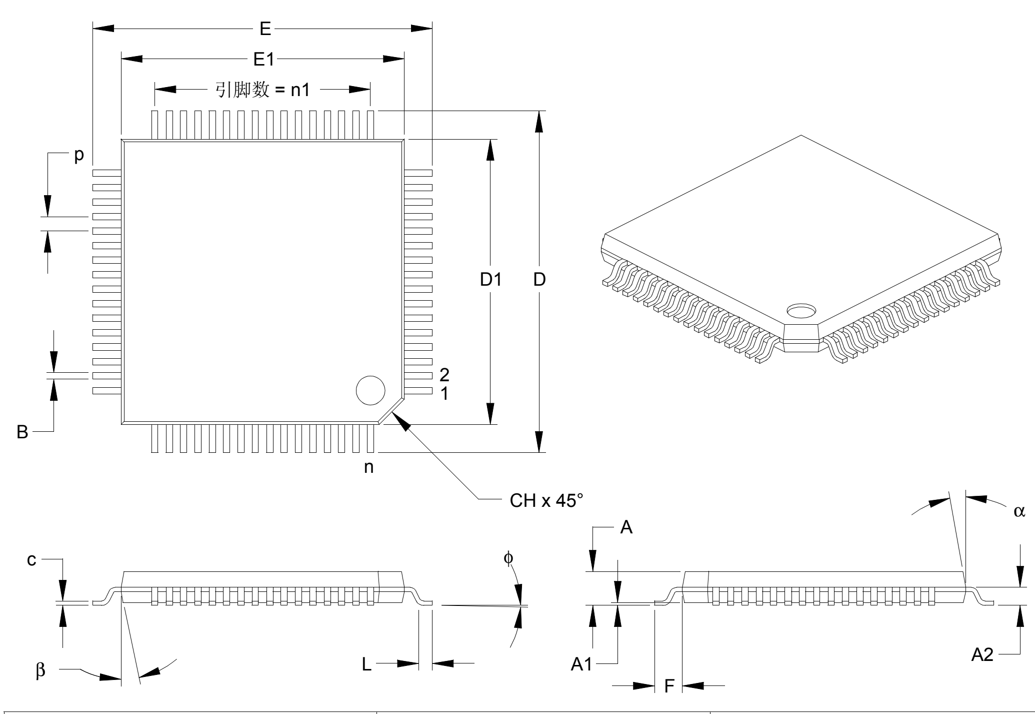
注： 如果 Microchip 器件编号不能在一行中完全标出，它将换行继续标出。因此限制了用户指定信息的可用字符数。

29.2 封装详细信息

以下部分将介绍各种封装的技术细节。

64 引脚塑封薄型正方扁平封装 （PT） 10x10x1 mm 主体， 1.0/0.10 mm 引脚形式 （TQFP）

注：	最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



单位		英寸			毫米 *		
尺寸范围		最小	正常	最大	最小	正常	最大
引脚数	n		64			64	
引脚间距	p		.020			0.50	
每侧引脚数	n1		16			16	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002	.006	.010	0.05	0.15	0.25
底脚长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度	F		.039 REF.			1.00 REF.	
底脚倾斜角	φ	0	3.5	7	0	3.5	7
总宽度	E	.463	.472	.482	11.75	12.00	12.25
总长度	D	.463	.472	.482	11.75	12.00	12.25
塑模封装宽度	E1	.390	.394	.398	9.90	10.00	10.10
塑模封装长度	D1	.390	.394	.398	9.90	10.00	10.10
引脚厚度	c	.005	.007	.009	0.13	0.18	0.23
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
引脚 1 切角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

* 控制参数

注

尺寸 D 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

REF: 参考尺寸，通常无公差，仅供参考。

参见 ASME Y14.5M

等同于 JEDEC 号: MS-026

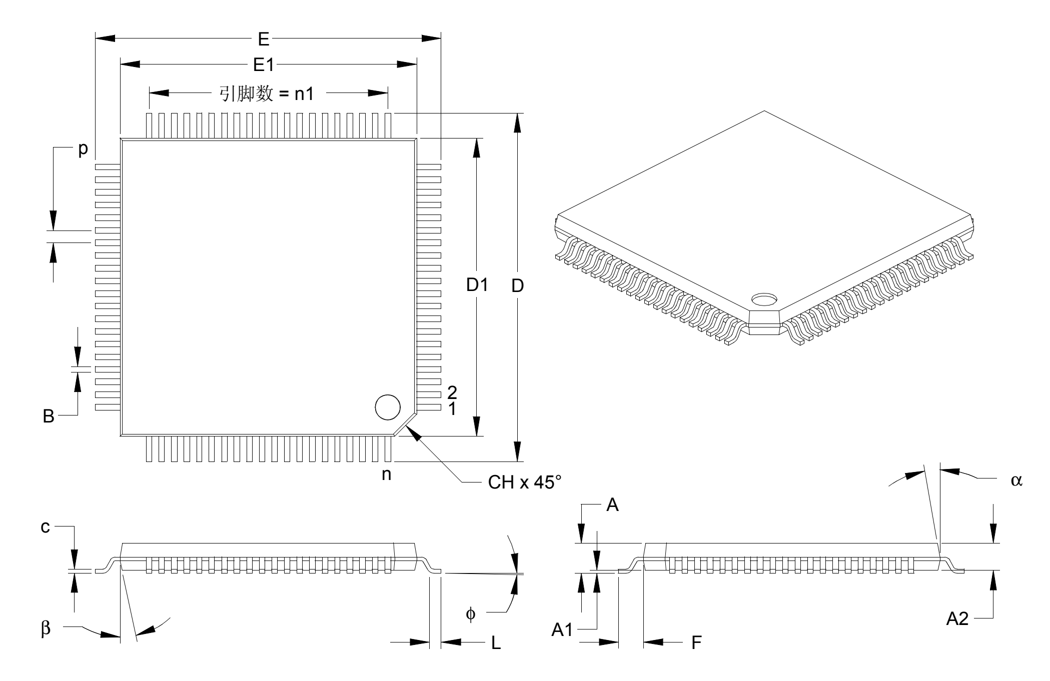
修订于 07-22-05

图号 C04-085

超前信息

80 引脚塑封薄型正方扁平封装 （PT） 12x12x1 mm 主体， 1.0/0.10 mm 引脚形式 （TQFP）

注：	最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



	单位	英寸			毫米 *		
	尺寸范围	最小	正常	最大	最小	正常	最大
引脚数	n		80			80	
引脚间距	p		0.020 BSC			0.50 BSC	
每侧引脚数	n1		20			20	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002	.004	.006	0.05	0.10	0.15
底脚长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度	F		.039 REF.			1.00 REF.	
底脚倾斜角	φ	0°	3.5°	7°	0°	3.5°	7°
总宽度	E		0.551 BSC			14.00 BSC	
总长度	D		0.551 BSC			14.00 BSC	
塑模封装宽度	E1		0.472 BSC			12.00 BSC	
塑模封装长度	D1		0.472 BSC			12.00 BSC	
引脚厚度	c	.004	.006	.008	0.09	0.15	0.20
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
引脚 1 切角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5°	10°	15°	5°	10°	15°
塑模底部锥度	β	5°	10°	15°	5°	10°	15°

* 控制参数

注：

尺寸 D1 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

BSC：基本尺寸。理论上显示的是没有公差的精确值。
参见 ASME Y14.5M

REF：参考尺寸，通常无公差，仅供参考。
参见 ASME Y14.5M

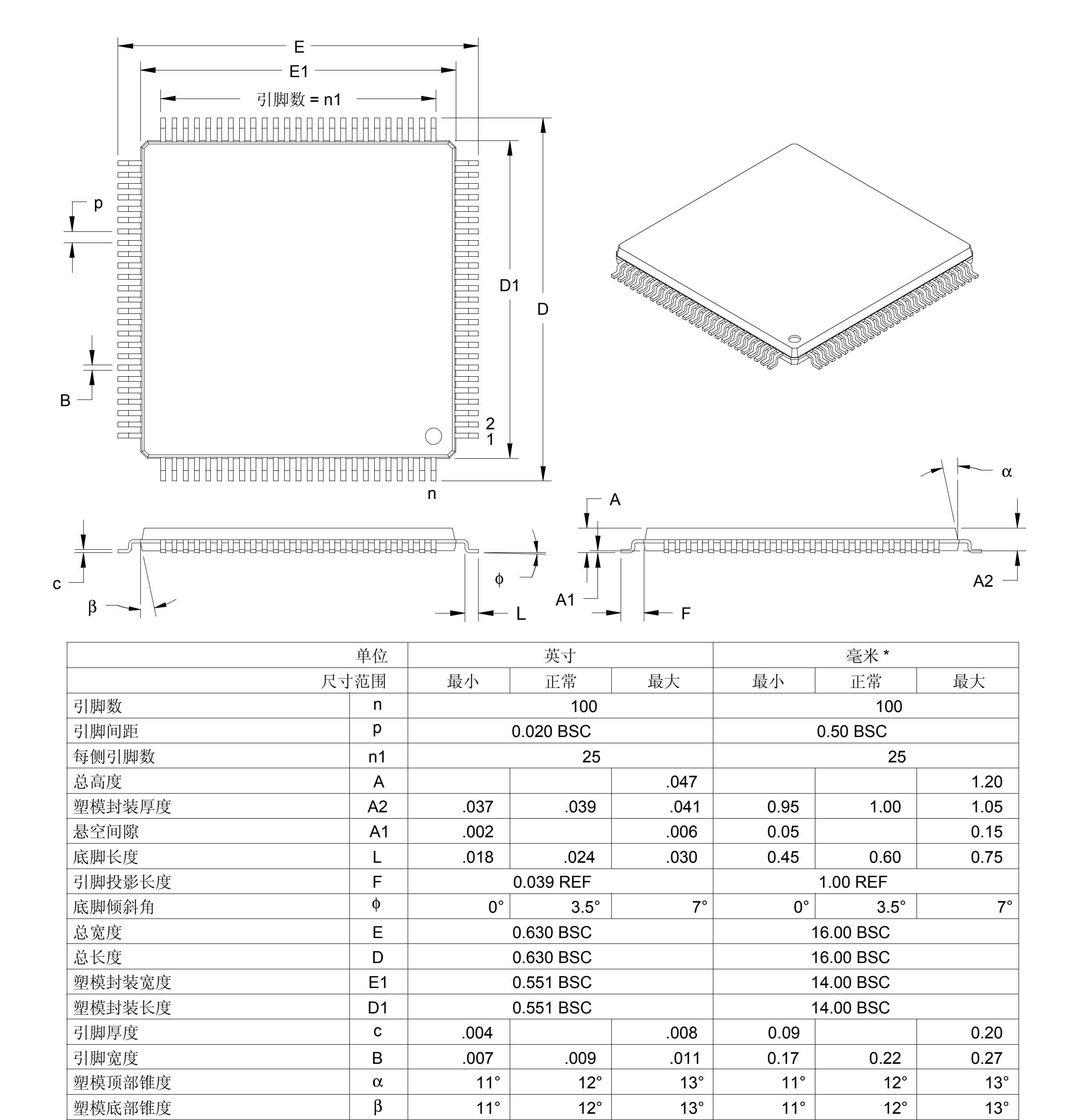
等同于 JEDEC 号：MS-026

图号 C04-092

修订于 07-22-05

100 引脚塑封薄型正方扁平封装（PF） 14x14x1 mm 主体， 1.0/0.10 mm 引脚形式（TQFP）

注：	最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



	单位	英寸			毫米 *		
	尺寸范围	最小	正常	最大	最小	正常	最大
引脚数	n		100			100	
引脚间距	p		0.020 BSC			0.50 BSC	
每侧引脚数	n1		25			25	
总高度	A			.047			1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002		.006	0.05		0.15
底脚长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度	F		0.039 REF			1.00 REF	
底脚倾斜角	φ	0°	3.5°	7°	0°	3.5°	7°
总宽度	E		0.630 BSC			16.00 BSC	
总长度	D		0.630 BSC			16.00 BSC	
塑模封装宽度	E1		0.551 BSC			14.00 BSC	
塑模封装长度	D1		0.551 BSC			14.00 BSC	
引脚厚度	c	.004		.008	0.09		0.20
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
塑模顶部锥度	α	11°	12°	13°	11°	12°	13°
塑模底部锥度	β	11°	12°	13°	11°	12°	13°

* 控制参数

注：

尺寸 D1 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

BSC：基本尺寸。理论上显示的是没有公差的精确值。
参见 ASME Y14.5M

REF：参考尺寸，通常无公差，仅供参考。
参见 ASME Y14.5M

等同于 JEDEC 号：MS-026

图号 C04-110

修订于 07-21-05

附录 A： 版本历史

版本 A （2006 年 3 月）

PIC18F97J60 系列器件的原始数据手册。

附录 B： 器件差异

表 B-1 为本数据手册中所列器件之间的差异。

表 B-1： PIC18F97J60 系列中的器件差异

功能部件	PIC18F66J60	PIC18F66J65	PIC18F67J60	PIC18F86J60	PIC18F86J65	PIC18F87J60	PIC18F96J60	PIC18F96J65	PIC18F97J60
程序存储器 （字节）	64K	96K	128K	64K	96K	128K	64K	96K	128K
程序存储器 （指令）	32764	49148	65532	32764	49148	65532	32764	49148	65532
中断源	26			27			29		
I/O 端口 （引脚）	端口 A, B, C, D, E, F, G (39)			端口 A, B, C, D, E, F, G, H, J (55)			端口 A, B, C, D, E, F, G, H, J (70)		
增强型 USART 模块	1			2					
MSSP 模块	1						2		
并行从动端口通信 （PSP）	无						有		
外部存储器总线	无						有		
10 位模数转换模块	11 路输入通道			15 路输入通道			16 路输入通道		
封装	64 引脚 TQFP			80 引脚 TQFP			100 引脚 TQFP		

注：

索引

A

B

C

 超前信息

H

I

J

K

L

超前信息

 超前信息

T

Z

超前信息

注：

MICROCHIP 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和样本程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及存档软件
- **一般技术支持**——常见问题（FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的客户通知服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 www.microchip.com，点击“变更通知客户（Customer Change Notification）”服务并按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师（FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过http://support.microchip.com获得网上技术支持。

读者反馈表

我们努力为您提供最佳文档，以确保您能够成功使用 Microchip 产品。如果您对文档的组织、条理性、主题及其他有助于提高文档质量的方面有任何意见或建议，请填写本反馈表并传真给我公司 TRC 经理，传真号码为 86-21-5407-5066。

请填写以下信息，并从下面各方面提出您对本文档的意见。

致: TRC 经理　　　　总页数 ________

关于: 读者反馈

发自: 姓名 ________________________________

公司 ________________________________

地址 ________________________________

国家 / 省份 / 城市 / 邮编 ________________________________

电话: (_______) _________ - _________　　　传真: (______) _________ - _________

应用（选填）:

您希望收到回复吗？是 ____ 否____

器件: PIC18F97J60 系列　　　文献编号: DS39762A_CN

问题

1. 本文档中哪些部分最有特色？

____

____

2. 本文档是否满足了您的软硬件开发要求？如何满足的？

____

____

3. 您认为本文档的组织结构便于理解吗？如果不便于理解，那么问题何在？

____

____

4. 您认为本文档应该添加哪些内容以改善其结构和主题？

____

____

5. 您认为本文档中可以删减哪些内容，而又不会影响整体使用效果？

____

____

6. 本文档中是否存在错误或误导信息？如果存在，请指出是什么信息及其具体页数。

____

____

7. 您认为本文档还有哪些方面有待改进？

____

____

产品标识体系

欲订货或获取价格、交货等信息，请与我公司生产厂或各销售办事处联系。

器件编号	X	/XX	XXX
器件	温度范围	封装	模板

器件	PIC18F66J60/66J65/67J60， PIC18F86J60/86J65/87J60， PIC18F96J60/96J65/97J60， PIC18F66J60/66J65/67J60T[1]， PIC18F86J60/86J65/87J60T[1]， PIC18F96J60/96J65/97J60T[1]；
温度范围	I = -40°C 至 +85°C （工业级）
封装	PT = 64 和 80 引脚 TQFP （薄型正方扁平封装） PF = 100 引脚 TQFP （薄型正方扁平封装）
模板	QTP、SQTP、代码或特殊要求 （其他情况空白）

示例：

a） PIC18F67J60-I/PT 301 = 工业级温度， TQFP 封装， QTP 模板 #301。

b） PIC18F67J60T-I/PT = 卷带式，工业级温度， TQFP 封装。

注 1： T = 卷带式



全球销售及服务网点

美洲

公司总部 **Corporate Office**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://support.microchip.com
网址：www.microchip.com

亚特兰大 **Atlanta**
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

波士顿 **Boston**
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 **Chicago**
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 **Dallas**
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 **Detroit**
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

科科莫 **Kokomo**
Kokomo, IN
Tel: 1-765-864-8360
Fax: 1-765-864-8387

洛杉矶 **Los Angeles**
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣克拉拉 **Santa Clara**
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

加拿大多伦多 **Toronto**
Mississauga, Ontario,
Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

亚太总部 **Asia Pacific Office**
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Habour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 北京
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 – 福州
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 顺德
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

中国 – 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 – 西安
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

台湾地区 – 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-572-9526
Fax: 886-3-572-6459

亚太地区

澳大利亚 **Australia - Sydney**
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 **India - Bangalore**
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

印度 **India - New Delhi**
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

印度 **India - Pune**
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 **Japan - Yokohama**
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 **Korea - Gumi**
Tel: 82-54-473-4301
Fax: 82-54-473-4302

韩国 **Korea - Seoul**
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 **Malaysia - Penang**
Tel: 60-4-646-8870
Fax: 60-4-646-5086

菲律宾 **Philippines - Manila**
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 **Singapore**
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 **Thailand - Bangkok**
Tel: 66-2-694-1351
Fax: 66-2-694-1350

欧洲

奥地利 **Austria - Wels**
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦 **Denmark-Copenhagen**
Tel: 45-4450-2828
Fax: 45-4485-2829

法国 **France - Paris**
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 **Germany - Munich**
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

意大利 **Italy - Milan**
Tel: 39-0331-742611
Fax: 39-0331-466781

荷兰 **Netherlands - Drunen**
Tel: 31-416-690399
Fax: 31-416-690340

西班牙 **Spain - Madrid**
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

英国 **UK - Wokingham**
Tel: 44-118-921-5869
Fax: 44-118-921-5820

12/08/06